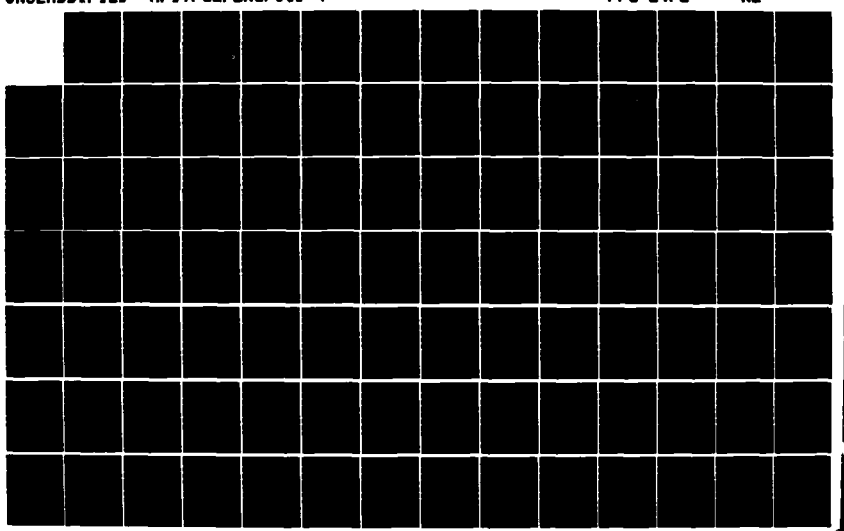
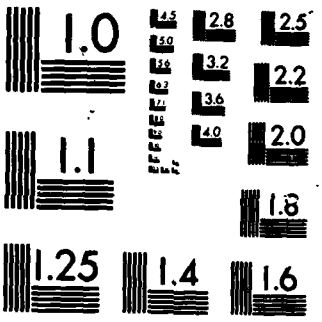
AD-A164 014 DESIGN OF A DIGITAL CONTROLLER FOR AN ULTRAGRAVIMETRIC MICROBALANCE(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING C J BOLAN DEC 85 1/2

UNCLASSIFIED AFIT/GE/ENG/85D-4 F/G 14/2 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

DESIGN OF A DIGITAL CONTROLLER FOR AN ULTRAGRAVIMETRIC MICROBALANCE

THESIS

Christopher J. Bolan
Second Lieutenant, USAF

AFIT/GE/ENG/85D-4

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GE/ENG/85D-4

DESIGN OF A DIGITAL CONTROLLER FOR AN ULTRAGRAVIMETRIC MICROBALANCE

THESIS

Christopher J. Bolan
Second Lieutenant, USAF

AFIT/GE/ENG/85D-4

Approved for public release; distribution unlimited

AFIT/GE/ENG/85D-4

DESIGN OF A DIGITAL CONTROLLER FOR AN ULTRAGRAVIMETRIC MICROBALANCE

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the degree of
Master of Science in Electrical Engineering

Christopher J. Bolan, B.S.
Second Lieutenant, USAF

December 1985

Approved for public release; distribution unlimited

Preface

The purpose of this project was to design and construct a digital control system for an ultragravimetric pivotal beam microbalance. The immediate need for this digital system is to replace the analog control circuitry presently controlling the microbalance in order to relieve the balance operator of various duties concerning the balance as well as improve the accuracy of the balance.

It was not possible to test the designed digital system on the microbalance itself, but, instead, the system was tested using circuitry to simulate the microbalance response. Although, this limited testing makes the results inconclusive, the tests did give encouraging answers. Therefore, correct operation of the balance with the designed system is predicted.

I am deeply indebted to my thesis advisor, Maj Donald Kitchen for his help, guidance, and support throughout this thesis effort. I would also like to thank my thesis reader, Maj Dale Hibner for his invaluable guidance during the design of the digital system and draft reviews. A word of thanks is also owed to Dr. Thomas Jones of the Avionics Laboratory for his help in analyzing circuit components.

Christopher J. Bolan

Table of Contents

List of Figures

List of Tables

Abstract

The design and construction of a digital control system for an ultragravimetric pivotal beam microbalance has been performed. The digital system replaces the present analog control circuitry of the microbalance. Included in the design is a hardware circuit, a Z-151 computer which has been interfaced to the circuit, and software to control the circuit and to operate the microbalance.

An analysis of the present analog control circuit is first presented. This analysis gives the overall function of the circuit as well as an indepth view of the configuration of a particular operational amplifier: the derivative compensator. The hardware for the digital control system is described followed by the development of the software which controls the system.

Studies performed on the experimental design revealed that the system satisfactorily simulates the analog control circuit. Tests show that the system automatically zeroes the balance, weighs the sample, stores the data onto a disk file, and terminates the experiment.

I. Introduction

The ultragravimetric pivotal vacuum microbalance is a highly responsive system which is capable of detecting very small changes in mass of samples (i.e. one-tenth of a gram per ten gram load). The microbalance is used to study topics such as the measurement of adsorption and desorption of gasses from solid surfaces (2:10; 11:711; 10:541; 3:1206).

The circuit which governs the operation of the balance is composed entirely of analog devices. That is, all of the components deal strictly with voltage levels. Using this "analog" circuit does present several disadvantages. For instance, the operator must continuously monitor the experiments in order to terminate the study as soon as the balance reaches equilibrium (i.e. the sample is balanced). Another disadvantage is that the operator must perform the tedious and lengthy operation of zeroing the balance before the study begins. Also, there is a section of the circuit, as noted later, which is extremely temperature sensitive, thus decreasing overall balance sensitivity when there are variations in the room temperature.

The objective of this thesis is to design and construct a digital system including a digital circuit, computer interface, and associated software to replace the analog control circuitry of an ultragravimetric pivotal microbalance. The designed digital system will perform much of the work which is currently done by the microbalance opera-

tor. This work includes zeroing the balance and terminating the experiment after it reaches equilibrium. Also, due to the fact that the designed system is composed of different circuitry, the temperature sensitivity section of the analog circuit is eliminated.

Results

A major result of this thesis is the successful design and construction of a digital control system which can replace the present control circuitry of the ultragravimetric microbalance. The control system designed includes a hardware circuit, Z-151 computer and hardware interface, and software program which is used for controlling the hardware.

The system is used to receive an error signal input from the microbalance infrared detectors, record the data, and output a current to the microbalance solenoid to produce a force to bring the microbalance into equilibrium. When initiated, the digital system automatically zeroes the balance, weighs the sample, and then stores the data to a disk file.

The sensitivity of the digital system is such that any changes in the error signal in excess of 2.45 microvolts are detected, and changes in the output are in 10 microvolt steps. Although the accuracy of the system is designed to be within this 10 microvolt step, the ground potential of the circuit decreases the accuracy by 20 microvolts.

The software to control the digital system is written in 8088 assembly language and is listed in Appendix E. The software reads the error signal from the microbalance via the hardware circuitry and controls the feedback to the microbalance by changing the output of the digital system. The software also stores the data into memory and then onto a disk file.

Another major result of this thesis is the documented analysis of the analog circuit presently used to control the microbalance operation. In this analysis, the overall function of the circuit is determined as well as an indepth view of a section of the circuit, the derivative compensator.

Assumptions

The following assumptions are taken into consideration in the design of the digital control system:

1. The maximum error signal from the microbalance infrared detectors is ten millivolts.

2. The balance response to the output of the digital system is complete before the next error signal sample is taken.

3. There is enough space in the Z-151's secondary memory (i.e. floppy disk) to store the data points.

4. The sample is heavier than the counterweight before the experiment.

5. The sample does not lose mass during the experiment.

Major Equipment

The major apparatus used for the control circuitry of the microbalance system is a Zenith Z-151 computer and its associated software which is capable of running assembly language programs. It is necessary to use assembly language programs in order to obtain the desired speed of operation and the capability to use I/O ports and interrupts. The program that is to run on the computer is the major element of the designed digital control circuit. The computer must also have a minimum of two 16-bit or four 8-bit parallel I/O ports. These ports are used to directly communicate with the digital control circuit via the input from the analog-to-digital converter (ADC) and the output to the digital-to-analog converter (DAC). Also used are various circuit components such as an ICL7109 ADC, HSDAC87 DAC, two AD525 high quality operational amplifiers (i.e. high linearity and low temperature drift), and test equipment. The above components are instrumental to the design of the digital controller.

Scope

This thesis is limited to the design, construction, and testing of a digital feedback control system which can replace the current analog circuit of the pivotal beam ultragravimetric vacuum microbalance. The designed feedback control system includes a circuit, its interface with the computer, and software for control of the system. The digital system is designed to be used with the existing error

signal detection circuitry (infrared sources, infrared detectors, and bridge) and the magnet and coil compensation circuitry.

The background description of the existing analog circuit is a generalized description of the function of major circuit components and does not describe each component in detail. This thesis does not implement the "new" system on a microbalance nor does it determine the calibration factor, which relates mass gain of the sample to the compensating signal. The software for the digital system is merely the basis of what is needed to control the circuitry and does not give a full analysis of the experiment. The digital circuit is a prototype and is left on an Elite 3 breadboard for further testing.

Approach and Presentation.

There are five primary phases that this thesis is divided into: a background literature search of the existing microbalance, the design and implementation of the digital circuit to replace the existing analog circuit of the microbalance, the computer interface for the digital circuit, the design and implementation of the software which controls the microbalance, and the testing of the system.

The background literature search which gives a general description of the existing microbalance system is presented in Chapter II. The parts of the microbalance system which are relevant to this thesis are explained first. Next, an

analysis of the analog circuit which controls the microbalance is given. It is necessary to determine the function of the analog circuit in order to design a replacement digital circuit.

In Chapter III, the design of the digital circuit is presented along with the components used in its implementation. Also included in this chapter is the hardware interface between the digital circuit and the computer.

Chapter IV presents the development of the software needed to analyze and control the activity of the microbalance. Included in this chapter is a description of what the software is to accomplish, flowcharts of the program, and a description of the program. The program listing is contained in an appendix.

The next section, Chapter V, discusses how the digital circuit, computer, and software were tested. This testing is necessary to determine whether the circuit will correctly give results by simulating the microbalance and the control circuit.

Finally, Chapter VI presents the results of the testing of the digital circuit and software. Also given, are the conclusions which are made from the testing results and any recommendations for further study on the implemented digital system.

II. Background

This chapter describes the components of the ultragravimetric pivotal vacuum microbalance which are of significance to the design of the digital feedback control system. The microbalance description is divided into two sections; the actual mechanism which conducts the physical balancing of a sample and the analog circuit which controls the balancing activity.

Balance Mechanism.

An ultragravimetric microbalance is shown in Figure II-1 (11:712). The ultramicrobalance housing (A) and the hangdown tubes (H) are where the actual balancing of the sample takes place. The components which are inside of the housing and hangdown tubes are described shortly. A parts legend for the balance shown in Figure II-1 can be found in Appendix A (11:712).

A diagram of a quartz pivotal beam balance is shown in Figure II-2 (2:34; 4:29; 9:79). It consists of a quartz beam resting on a fulcrum with weights hanging from both sides of the beam. The weight on one side is the sample that is being analyzed and the weight on the other side is the counterweight, called the tare. On the same side of the counterweight is a magnet and coil (or solenoid) which are discussed later. Although not seen in Figure II-2, there are gold flags attached to the ends of the beam which are used to

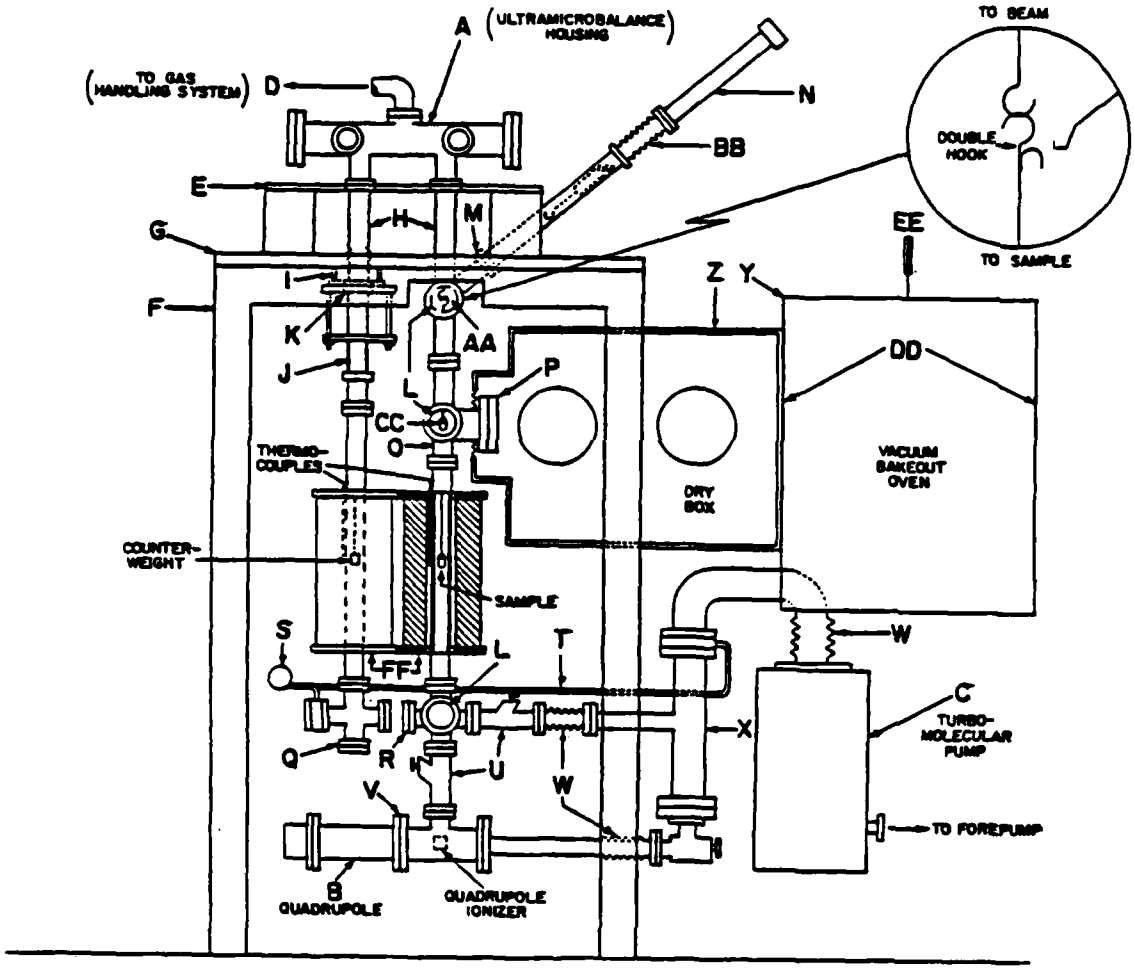


Figure II-1. Ultragravimetric Pivotal Vacuum Microbalance. See Appendix A for Legend of Parts (11:712)

detect beam deflection (whether the beam is balanced). These are described later.

For the reader to understand what type of feedback-control system is needed, it is necessary to explain how the balance detects beam deflection and then how the balance compensates for the beam deflection.

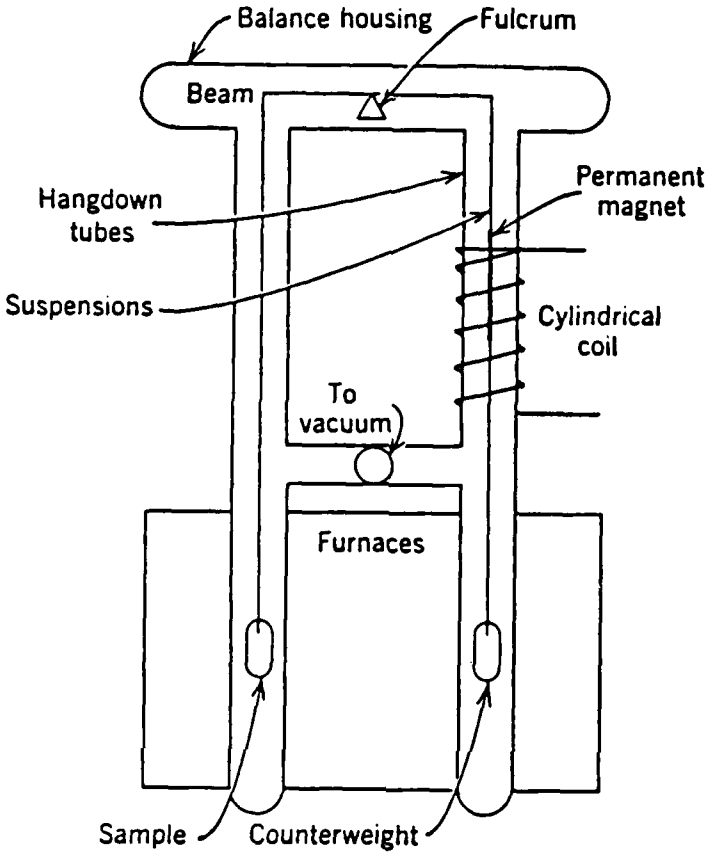


Figure II-2. Quartz Pivotal Beam Balance (2:34; 4:29; 9:79)

Beam Deflection. Beam deflection is optically detected through the use of infrared (IR) light sources, gold flags, and photodetectors. The alignment of these three components is shown in Figure II-3 (5:557).

The flags are constructed on a quartz window so that the top half of the window is transparent and the bottom half is covered with a thin layer of gold. The gold does not allow light, or radiation, to pass through the bottom half of the window. Hence, when the infrared light sources are incident

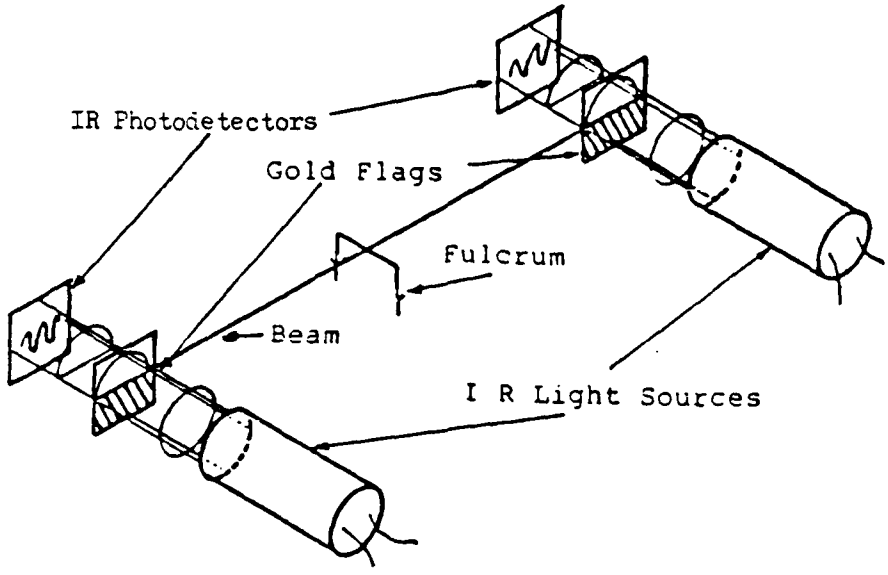


Figure II-3. Dual-path Light Source Apparatus for Detecting Beam Deflection of a Microbalance (5:557)

upon the window, radiation passes only through the top half of the window and is detected by IR detectors located behind each window.

When the quartz truss is in an equilibrium position, both detectors will receive the same intensity from the infrared light sources. However, when the balance is tilted (i.e. not in equilibrium), the detector on the side of the beam which is raised will detect less radiation than the detector on the lower side of the beam (i.e. the gold flag on the raised side of the beam will also be raised, blocking some of the radiation which is incident upon the window and the gold flag on the lower side of the beam will be lowered

and allow more radiation to pass through the window). Since the detectors on each side of the beam receive different amounts of radiation, there is a corresponding difference in the current each produces. This difference in current, called the error signal, is received by the analog circuit for both analysis and to provide a basis for a feedback signal which the analog circuit produces to force the balance back into equilibrium.

Stabilizing Force. As mentioned earlier, there must be a stabilizing force used to keep the microbalance in its equilibrium position. A magnet and solenoid, both located on the tare side of the beam, supply the restoring force.

As is shown in the pivotal beam balance of Figure II-2, a magnet attached to the suspension is surrounded by a solenoid. By forcing a current through the solenoid, a magnetic field is produced within the solenoid as shown in Figure II-4 (8:143). This magnetic field applies a vertical force to the magnet and either raises or lowers the magnet, depending on the direction of the current through the solenoid. The vertical force on the magnet is the compensating force which is used to keep the balance in its equilibrium position.

The feedback circuit including IR detectors, solenoid and magnet, and control circuit is shown as a block diagram in Figure II-5. The IR detectors produce an error signal which is input to the control circuit. The error signal is processed by the control circuit and a compensating signal

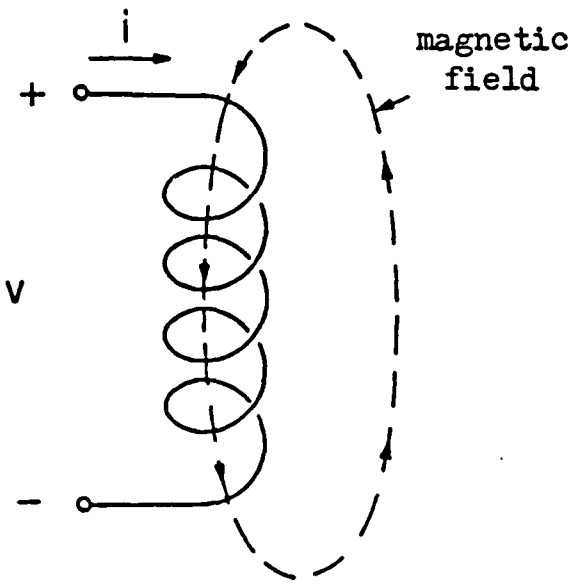


Figure II-4. Magnetic Field of a Solenoid (8:143)

is produced. This new signal is then input to the solenoid on the balance mechanism. The affect of the solenoid upon the balance causes the IR detectors to produce a new error signal. This feedback process continues until the operator intervenes.

ANALOG CIRCUIT.

A diagram of the analog circuit currently being used in the microbalance described above is given in Figure II-6 (12:1). A description of the functional parts is explained so that the reader can understand how the circuit operates.

The error signal that is input to the analog circuit is directly proportional to the current produced by the IR detectors previously described. Due to the resonant

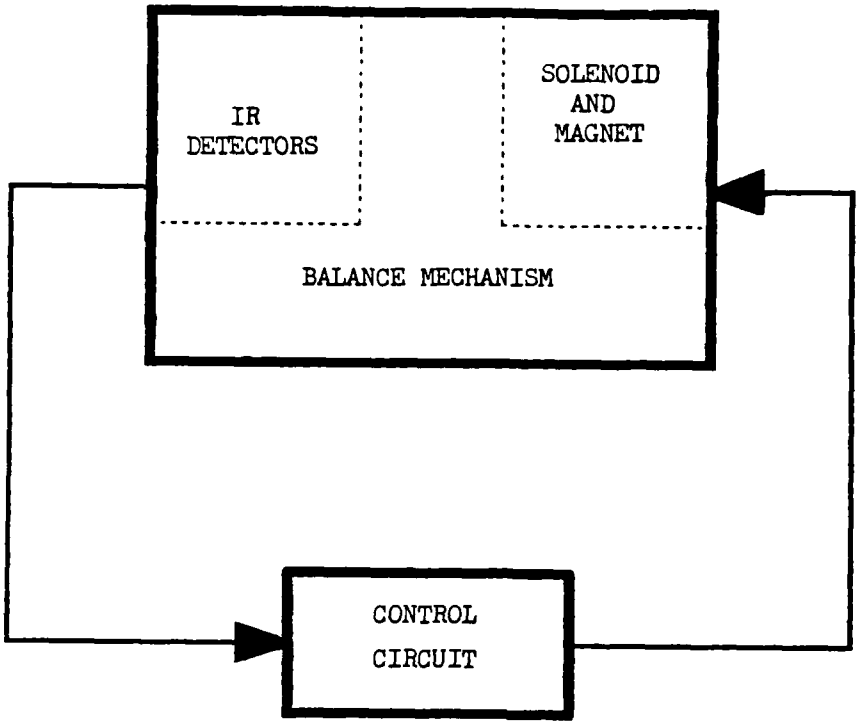


Figure II-5. Block Diagram of the Microbalance Feedback Circuit

frequency of the microbalance, the beam which rests on the pivot oscillates at a frequency of 1.76 Hz when the balance is in its equilibrium state (12:2). These oscillations show up in the signal that is sent to the analog circuit. Since this frequency is so small, it acts almost as a DC signal and, therefore, is very hard to filter out without affecting the DC signal. This means that the circuit must compensate for both the DC error signal and the oscillations.

Bridge Circuit. The photodiodes in the bridge circuit are the IR detectors that were shown on the ends of the balancing

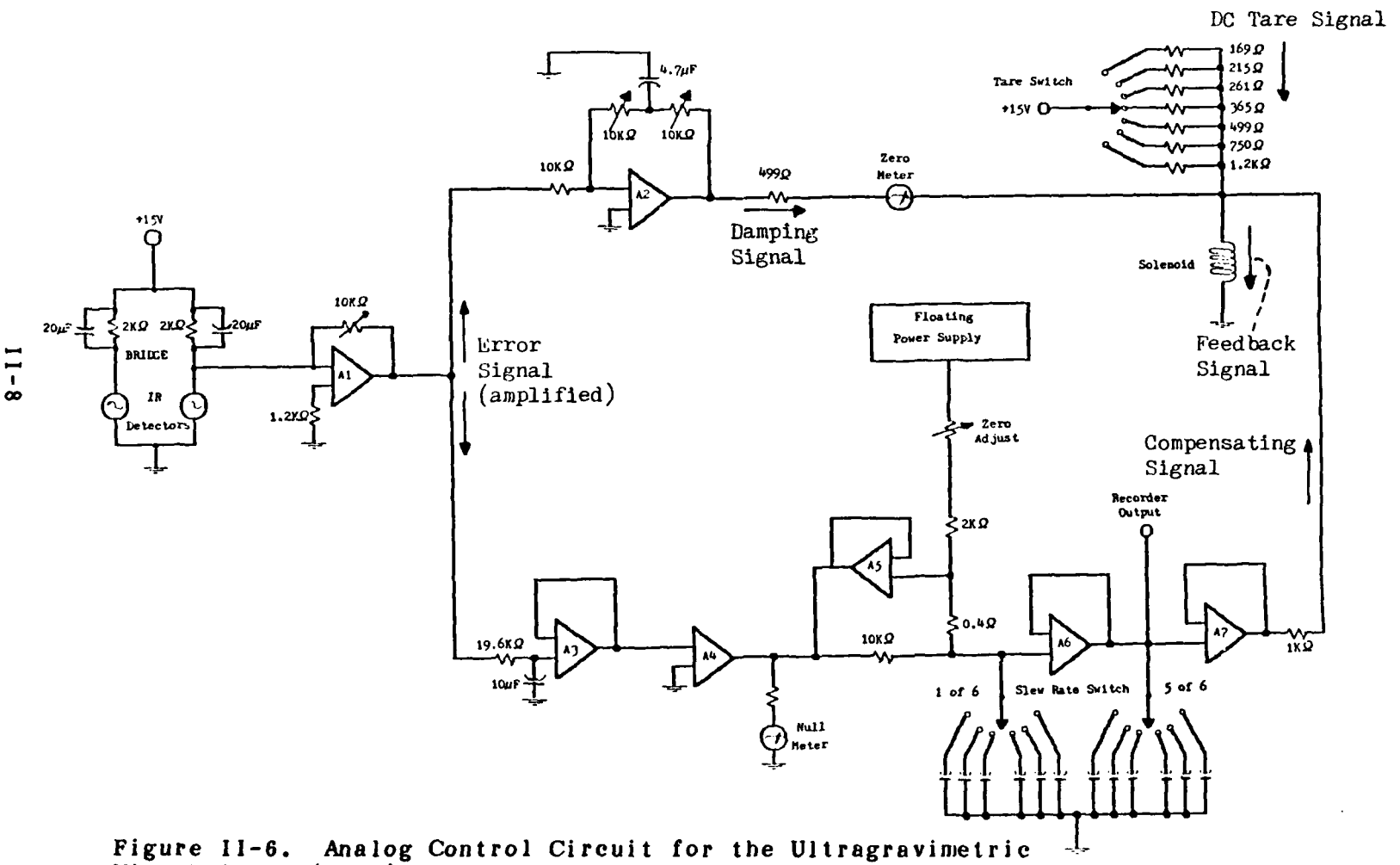


Figure II-6. Analog Control Circuit for the Ultragravimetric Microbalance (12:1)

beam of the microbalance in Figure II-3. The bridge circuit is a sensitive means for detecting a difference in the current supplied by each photodetector. The difference in current between the photodiodes is the error signal that is input to operational amplifier A1.

Op Amp A1. The error signal from the bridge which enters A1 is amplified to a larger voltage level so that any minute changes in the error signal will also be amplified. An advantage to working with an amplified signal is that noise introduced to the system will have a smaller impact upon the accuracy of the system (i.e. 1 microvolt of external noise has less impact on a 1 volt signal than it does on a 1 millivolt signal). The values of the resistors connected to A1 give it a gain of 8.33. The amplified signal is then input to op amps A2 and A3.

Op Amp A2. A2 is configured as a derivative compensator circuit. The transfer function for this configuration is derived in Appendix B. A2 reacts to both the magnitude of the error signal entering and its rate of change. The signal leaving A2 is called a damping signal because it is out-of-phase with the compensating signal produced by the other half of the analog circuit (see Figure II-6). The damping signal combines with the compensating signal at the solenoid for the required feedback. The purpose of this damping signal is to prevent the balance from going unstable. Moreover, since the A2 circuit reacts to the rate of change of the error signal,

the circuit anticipates any quick changes in the error signal to keep the balance from going unstable. For example, if the tare weight were to start rising very quickly, the derivative compensator circuit would sense the rate of change and increase the out-of-phase signal to dampen the motion of the balance.

Note further that the use of A2 as a derivative compensator imports a problem of noise upon the analog system. Since the A2 circuit reacts to the rate of change of the error signal, any spurious signal introduced in to the error signal is magnified (6:348). Hence, the balance reacts to unwanted noise.

Op Amp A3. Before the amplified signal reaches A3, noise greater than 5 Hz is filtered out by a resistor-capacitor low pass filter. Op amp A3 then acts as a buffer, isolating the signal from A1 and A2.

Op Amp A4. This amplifier is a full gain amplifier which takes the sinusoidal output of A3 (sinusoidal because of resonant frequency described earlier) and converts it to a square wave as in Figure II-7 (12:2). When the balance is in equilibrium, the input to A4 is a sinusoidal wave with no DC offset as in Figure II-7a. In this case, the output of A4 is a square wave with X = Y shown in Figure II-7b, where X and Y are the time periods that the signal is positive and negative valued, respectively. When the balance is not in equili-

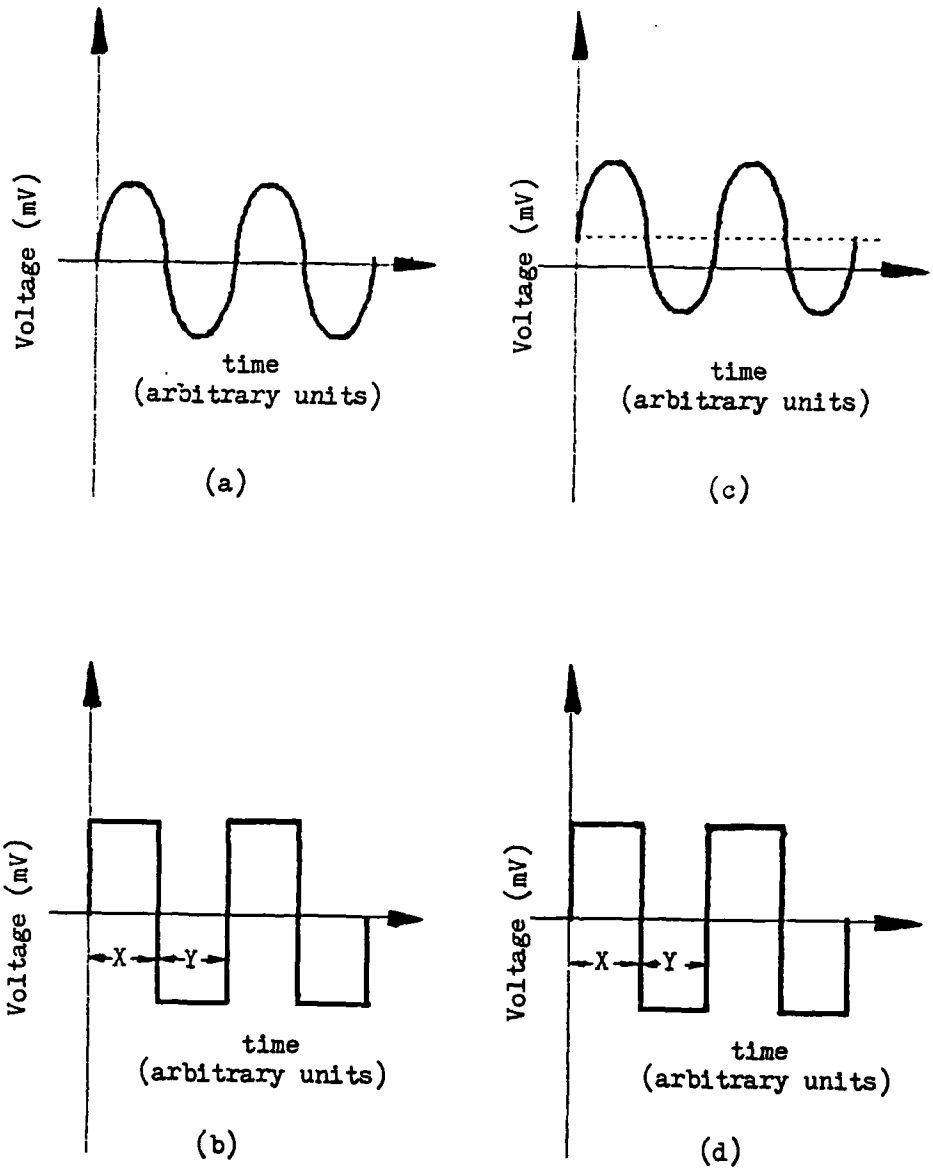


Figure II-7. Error Signal When Balance is in Equilibrium (a) Before Op Amp A4, and (b) After Op Amp A4. Error Signal When Balance is not in Equilibrium (c) Before Op Amp A4, and (d) After Op Amp A4 (12:2)

brium, there is a DC offset to the input of A4, as in Figure II-7c and, thus, $X \neq Y$ as shown in Figure II-7d.

The needle of the null meter at the output of A4 swings back and forth, spending an equal amount of time on each side if the balance is in equilibrium since the output is a square wave centered on the x-axis. When the balance is not in equilibrium, the needle will spend more time on one side than the other. The square wave signal then proceeds to a zeroing circuit containing A5, and then on to A6.

Op Amp A5 and Associated Circuitry. The purpose of A5 and its associated circuitry is to zero the balance and to provide the compensating signal for the balance and strip chart recorder. It is a feedback circuit and is correlated to the floating power supply which is shown in Figure II-6. The power supply uses the square wave signal from A4 as a floating ground. The voltage divider between the power supply and the floating ground is adjusted by the fine zero and coarse zero potentiometers. These potentiometers are adjusted before the experiment and are used to bring the balance into its equilibrium position. The coarse zero potentiometer is used to bring the balance across the null position and the fine zero potentiometer is used to fine tune the balance so that it is at the null position.

Op Amp A6 and Capacitor Banks. The capacitor banks before and after A6 are used to adjust the slew rate, or the rate of change of the voltage of the square wave signal. The proper

slew rate is chosen by varying the slew rate switch, thus selecting different combinations of capacitors from the capacitor banks. The capacitors are also used to integrate the square wave signal output from A4, providing a voltage level approximately equal to the area under the square wave. This voltage level in addition to the signal produced by the A5 circuitry is the compensating signal which is sent to the solenoid. Since the compensating signal is proportional to the mass change of the sample in the microbalance (3:1208), the signal at the output of A6 is recorded by a strip chart recorder, marked "Recorder Output" in Figure II-6, for analysis of data.

Op Amp A7 and Resistor. A7 is a buffer which isolates the compensating signal that goes to the solenoid from the rest of the analog circuit. The function of the resistor which follows A7 is so that the voltage output from A7 is dropped across the resistor rather than across the solenoid. Recall, it is current through the solenoid which induces a magnetic field, not voltage.

Tare Switch. This switch is used to balance out the sample and counterweight as much as possible. If the sample is significantly heavier than the counterweight, the tare switch is used. By setting the switch to the proper value, an additional supply of current is sent to the solenoid, creating a vertical force on the magnet. This vertical force provides an offset which is approximately equal to the

difference in weight between the sample and counterweight. In other words, the tare switch acts as a coarse adjustment for zeroing the balance.

Solenoid. The intersection at the top of the solenoid is where the compensating signal is combined with the damping signal and the constant DC signal produced by the tare switch. These signals combined together form the feedback signal.

From the previous analysis it is important to see the overall operation of the circuit in order to design and construct its replacement. The main items to realize are 1) there is a very small error signal entering the circuit, 2) the signal is then processed and the data is recorded, and 3) a feedback signal is sent back to the balance in an attempt to bring the balance to equilibrium.

III. Circuit Design

The purpose of this Chapter is to describe, in detail, a feedback system which replaces and improves upon the analog control circuit described in Chapter II. From the description of the microbalance system, recall that the microbalance control circuit must possess, as a minimum, the following characteristics: 1) receive a very small voltage (millivolts), 2) record the voltage and time of data continuously, and 3) output a very small current which will compensate for the input voltage. That is, drive the input voltage (error signal) to a zero reading.

With these characteristics in mind, a block diagram of a feedback control circuit was designed and is shown in Figure III-1. A general description of the concept behind the design is as follows:

The bridge produces an error signal from the IR detectors, as in the circuit of Chapter II, which is amplified by op amp B1 to a voltage level large enough to be converted to a digital signal by the analog-to-digital converter (ADC). The digitalized error signal is then input into a computer for analysis and generation of a compensating signal. After analysis of the error signal, the computer outputs a digital compensating signal to the digital-to-analog converter (DAC), which is converted to an analog compensating signal. The compensating signal is then attenuated by a voltage divider to a signal that is of approximately the same magnitude as

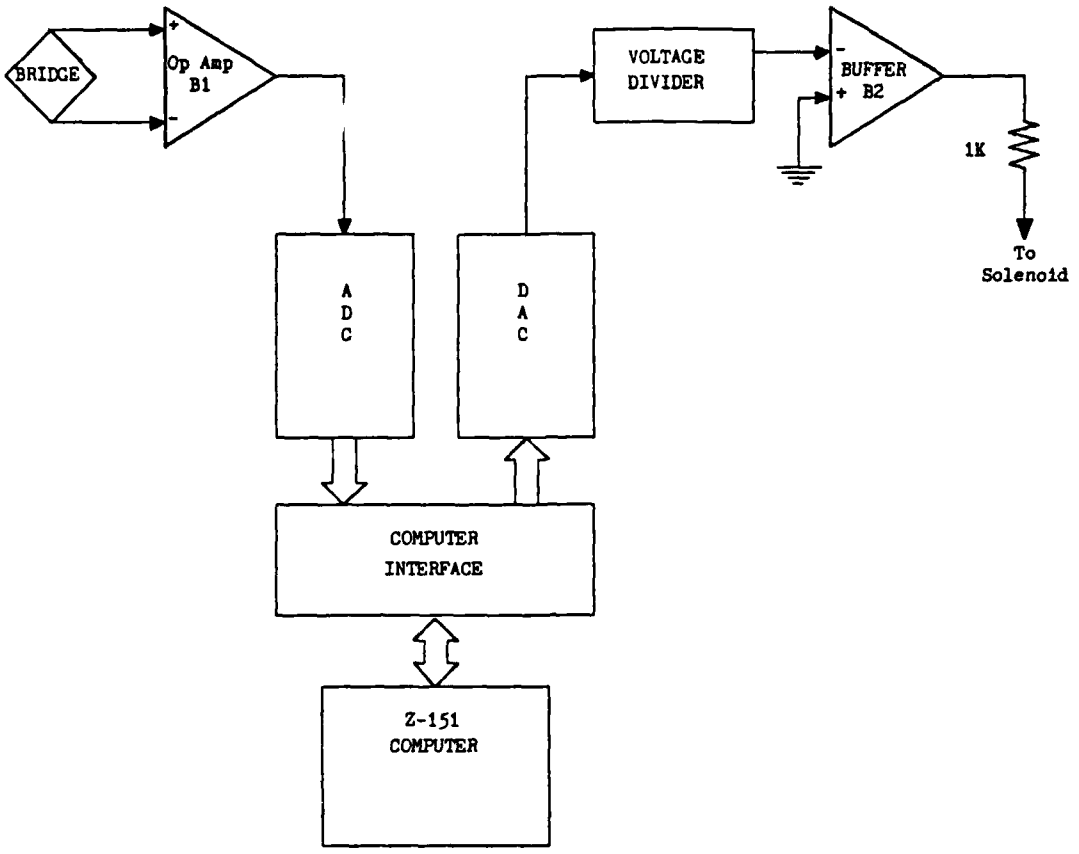


Figure III-1. Block Diagram of Digital Feedback System for the Microbalance

the input signal. The signal is then isolated from the circuit and applied to the solenoid of the microbalance. This process continues until the solenoid produces the counterbalancing force which places the balance in the equilibrium position and sends a zero error signal to the control circuit.

The remainder of this chapter includes a detailed description of the major components of the system described above, followed by a description of the interface between the computer and the circuit.

Detailed Hardware Design.

Using the block diagram of Figure III-1, selected components for the circuit are described as follows:

Bridge. As in the previously described circuit in Chapter II, the bridge is necessary to accurately detect the differential voltage between the two photodetectors. Recall, this voltage ranges from zero to approximately ten millivolts. The differential voltage which is detected is the error signal which will serve as an input to the rest of the circuit.

Op Amp B1. This op amp is an AD524 instrumentation operational amplifier. This particular op amp is used because it possesses the high sensitivity and linearity that is needed for the detection of minute voltage changes. A schematic of its connections to the circuit is shown in Figure III-2 (14:3).

Assuming a maximum input voltage (error signal) of 10 millivolts (mV), and a desired corresponding maximum output voltage of 3 volts (V), the voltage gain, G, is therefore

$$(3\ V)/(10\ mV) = 300.$$

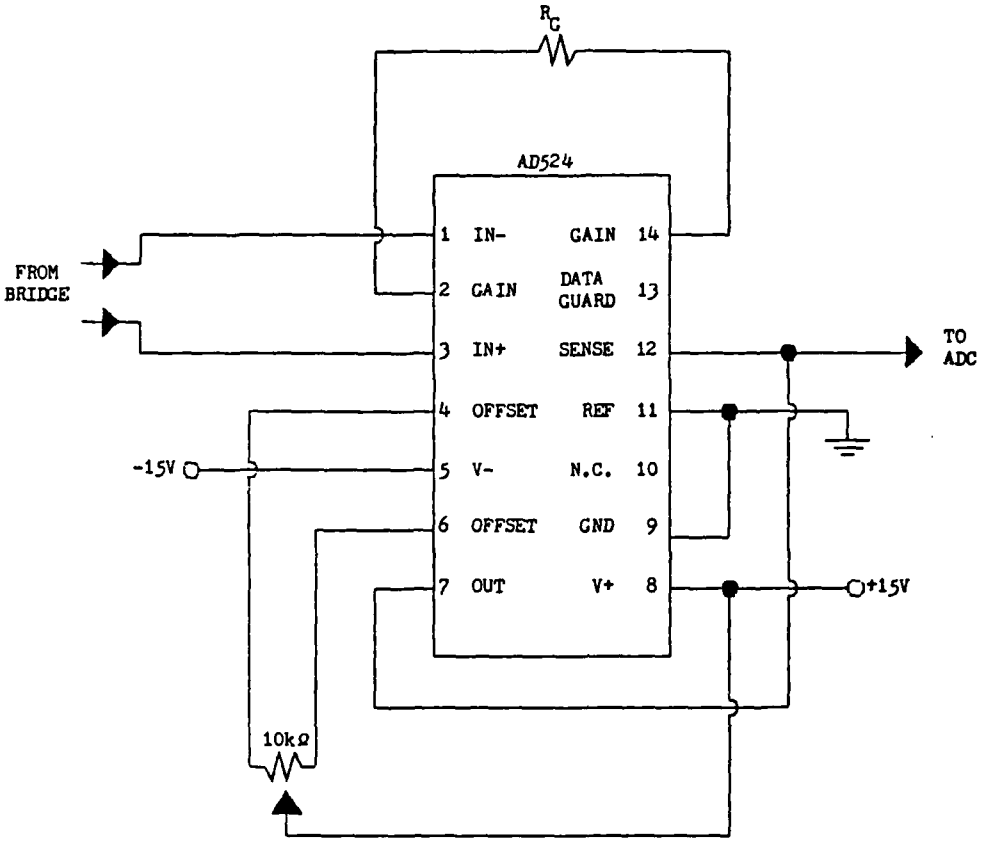


Figure III-2. Connections for Op Amp B1 in the Circuit (14:3)

The gain resistor, RG is computed by the following equation:

$$RG = 200k/(G - 1) \text{ ohms} \qquad (14:3)(1)$$

$$= 200k/(300 - 1) \text{ ohms}$$

$$= 669 \text{ ohms}.$$

Since the AD524 is very sensitive, it is significantly affected by environmental parameters, such as temperature. It is, therefore, suggested that R_G have a temperature coefficient less than 5ppm/°C (14:3).

The output of the AD524 is then input to the analog-to-digital converter.

Analog-to-Digital Converter (ADC). The ADC used is the ICL7109. This particular ADC has 12-bit accuracy, as opposed to the standard 8-bit accuracy. It is used for this circuit because of its high accuracy, drift of less than 1 μV/°C, low noise, and its compatibility with a microprocessor. Its purpose is to convert the analog input signal from the AD524 into a digital form so that it can be read by a digital computer. The pin connections from the ADC to the rest of the circuit are shown in Figure III-3. A brief description of the function of each pin is given in Appendix C (7:4-44).

Using power supplies of ± 5 V, the input voltage (amplified error signal from op amp B1), V_{IN}, must have a range of $-3.5\ V < V_{IN} < 4.0\ V$ (7:4-42). As a safety margin, the actual input voltage upon the ADC was chosen to be $-3.0\ V < V_{IN} < 3.0\ V$. With a max V_{IN} of 3 V, the voltage reference, V_{REF}, is determined by

$$V_{REF} = (1/2) \cdot V_{IN(MAX)} \qquad (7:4\text{-}35)\ (2)$$
$$= (1/2) \cdot 3\ V$$
$$= \underline{1.5\ V}.$$

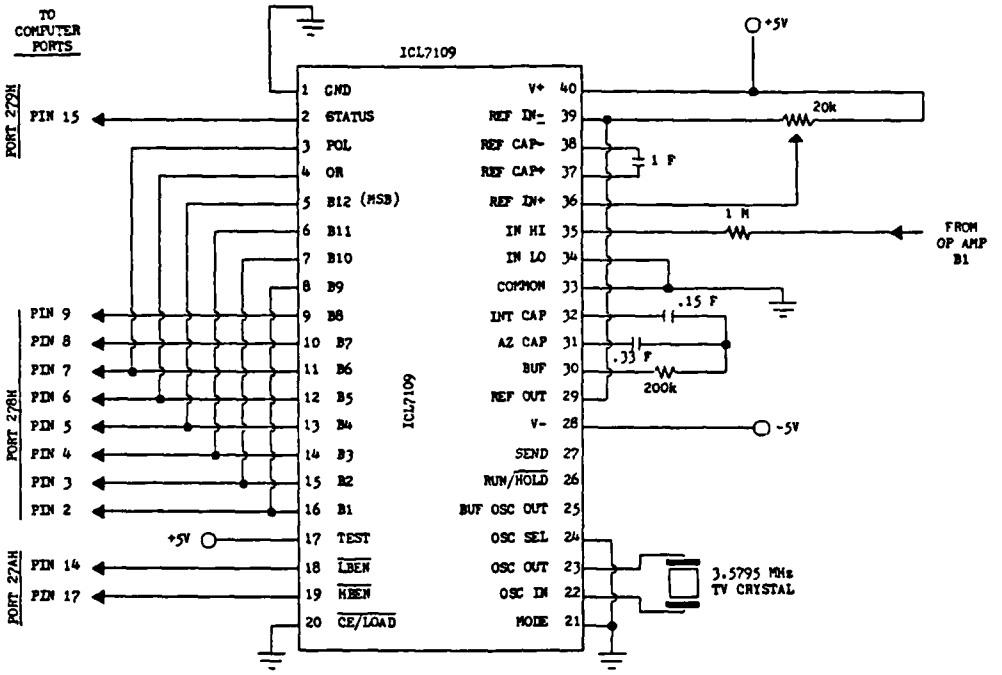


Figure III-3. ADC Pin Connections to the Circuit (7:4-41)

With this reference voltage, a voltmeter was placed between REF IN+ (pin 36) and REF IN- (pin 39) and the potentiometer connected to REF OUT (pin 29) was adjusted to give a VREF= 1.5 V.

The crystal connected to pins 22 and 23 is a 3.58 MHz TV crystal and gives the ADC a conversion rate of 7.5 conversions per second (7:4-30). Although the ADC is capable of handling a 245.8 MHz crystal which gives a conversion rate of

30 per second, the conversion rate using the TV crystal is sufficient for this particular application.

The status line, pin 2, outputs the status of the ADC. When this pin is high, the ADC is in a conversion phase and is still in the process of converting the input to a digital output. As soon as the conversion is complete and the data is latched to the output of the ADC, the status line goes low. The status line will stay low until the ADC resets itself for conversion of new data. Thus, the time to sample the ADC is when the status line is on a high-to-low transition so that only one sample is taken per conversion.

Since the ICL7109 ADC is microprocessor compatible, it was designed to output eight bits at a time (most computer ports are limited to eight bits of data.) The selection of which bits of the 12 ADC bits are placed on the output lines is determined by pins 18 and 19, LBEN* and HBEN*, respectively. When LBEN* is set low and HBEN* is set high, the lower order bits, B1-B8, of the ADC are activated and the other output bits are disabled. When LBEN* is set high and HBEN* is set low, then the upper four bits, B9-B12; POL, the polarity bit; and OR, the overrange bit, are activated and B1-B8 are disabled. This allows us to "tie" the output lines together as shown in Figure III-3. Using this configuration, it is necessary for the microprocessor to be able to control LBEN* and HBEN* so that it can channel the correct data on to its data lines.

Digital-to-Analog Converter (DAC) and Latches. The DAC used in this application is the HSDAC87. This DAC is chosen because of its high accuracy (12 bits) and low drift voltage (3 ppm/^{o}C) (15:73). Its purpose is to convert the digital signal from the computer to an analog signal that can be applied to the microbalance. The pin connections between the DAC and the rest of the circuit are shown in Figure III-4 (12:1). The configuration shown was chosen in order to get a output range of 0 to +5V (15:74). Using this output range, the voltage steps of the DAC are calculated as follows:

$$\text{Volt/Step} = (5\ \text{V})/(2^{12}\ \text{Steps})$$
$$= .00122\ \text{V}$$
$$= 1.22\ \text{mV}.$$

Since the microbalance performs in the millivolt range, it would be beneficial for the increment to be much smaller than 1 millivolt so that higher accuracy may be obtained (i.e. if the signal from the DAC is used as the output to the solenoid, the accuracy would only be within 1.22 mV of the desired output). Assuming that an increment of 10 microvolts (μV) supplies sufficient accuracy, a voltage divider is placed between the DAC and the circuit to reduce the DAC's output from 1.22 mV to about 10 μV, giving a gain of $\approx$ 0.01.

Since the DAC does not have an enable switch on it, latches must be interfaced between the DAC and the computer. The reason for the latches is that the computer can only send out eight bits of data at a time and the DAC is a 12 bit device. This means that as soon as the computer outputs the

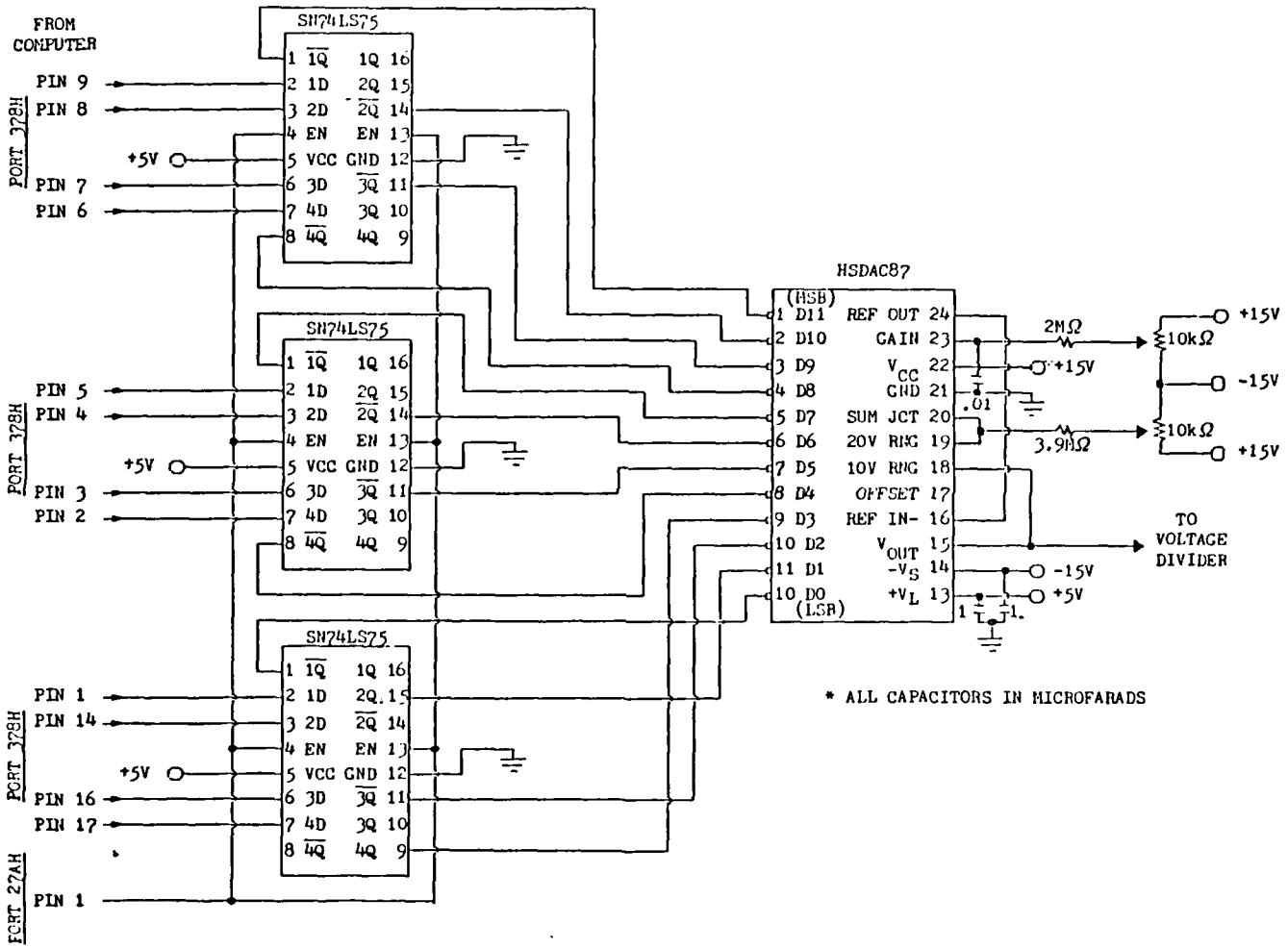


Figure III-4. DAC Pin Connections to the Circuit (15:74)

lower eight bits of data, the DAC will start to convert all 12 bits of data to an analog voltage if there were no latching devices present. To prevent this from happening, "latches" are used so that all 12 bits of data are enabled at the DAC at the same time. As soon as the latches have received all 12 bits of data, the computer sends a signal to enable the data to be latched to the DAC. The latches used in this application are SN74LS75's. Their pin connections to the circuit and the computer are shown in Figure III-4. Since the data inputs to the DAC are all active-low, the inverting outputs of the latches are used so that the computer works with all active-high signals.

Voltage Divider, Buffer, and Resistor. As noted earlier, the gain of the voltage divider should be approximately equal to 0.01. Using a 13 ohm and a 1500 ohm resistor as in Figure III-5 (14:3), the voltage gain, G, is calculated to be

$$G = (13 \text{ ohms})/(1500 \text{ ohms})$$
$$= 0.0087.$$

The signal from the DAC is reduced by the voltage divider, producing the compensating signal. The reduced signal is isolated by the inverting buffer, as shown in Figure III-5. The voltage of the signal is then dropped across the resistor and the resulting current flows through the solenoid, producing the compensating force. The buffer

chosen for this task is also an AD524 (same as op amp B1) due to its high sensitivity.

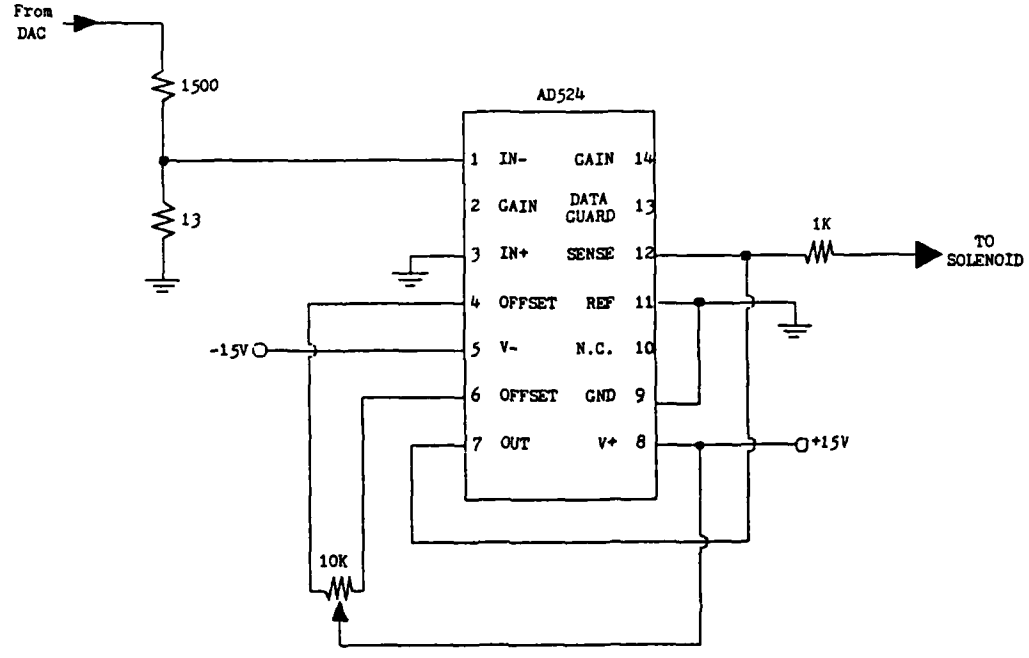


Figure III-5. Voltage Divider, Buffer and Resistor (14:3)

Computer Interface.

The computer used in this design is a modified Zenith Z-151 personal computer. The modifications include the addition of 320 kB of RAM memory for a total of 640 kB of RAM memory, and the addition of the Apparat Combo II card which provides the computer with an additional parallel printer port used for communicating with the control circuit previously described in this chapter. The Combo II card was modified so that its parallel port has bidirectional data flow rather than the originally intended "output only" data flow. This modification is shown in Appendix D.

With the addition of the Combo II card, the computer has access to two parallel printer ports. Each of these printer ports contain four memory addressable data ports which range from four to eight bits each. A bit map of the three parallel ports used for this application is shown in Table III-1 (13:9-10). The port numbers shown in Table III-1 are the addresses that are used in the software routine to specify the particular port. Ports 378H, 379H, and 37AH are the addressable ports which make up the parallel printer port originally in the computer, whereas the parallel printer port on the Combo II card is composed of ports 278H, 279H, and 27AH.

TABLE III-1. Parallel Port Bit Map Table (13:9-10)

PORT 378H/278H (Input/Output)

DATA BIT CONNECTOR	PRINTER	25 PIN
D0	data bit 0	pin 2
D1	data bit 1	pin 3
D2	data bit 2	pin 4
D3	data bit 3	pin 5
D4	data bit 4	pin 6
D5	data bit 5	pin 7
D6	data bit 6	pin 8
D7	data bit 7	pin 9

PORT 379H/279H (Input)

DATA BIT CONNECTOR	PRINTER	25 PIN
D0	unused	
D1	unused	
D2	unused	
D3	error	pin 15
D4	(-)select	pin 13
D5	(-)pe	pin 12
D6	ack	pin 10
D7	busy	pin 11

PORT 37AH/27AH (Output)

DATA BIT CONNECTOR	PRINTER	25 PIN
D0	(-)strobe	pin 1
D1	(-)auto lf	pin 14
D2	init	pin 16
D3	(-)slct in	pin 17
D4	IRQ enable	
D5	disable I/O drivers (optional)	
D6	unused	
D7	unused	

(-) indicates an active low signal (inverted before reaching the CPU.)

There are twenty-four information lines which must be connected between the computer and the circuit. These lines are shown in Figures III-3 and III-4 and are summarized as follows:

8 data input lines from ADC
1 status line input from ADC
2 data input control lines from ADC
12 data output lines to latches
1 enable line to latches
--

24 total connections.

Shown in Table III-2 are the corresponding pin assignments from the computer ports to the control circuit. Since the DAC has active-low inputs, most of the data signals from the computer to the DAC will be inverted by the latches before arriving at the DAC. The pins assigned to the DAC which correspond to the inverted signals are marked, in Table III-2, by *Latched with the exception of D1 and D3 of port 37AH. These two pins, D1 and D3, are not inverted by the latches because they are internally inverted by the computer hardware. Note also that D5 of port 27AH is set high. This pin corresponds to a modification that was made in the Combo II card and when set high configures the port to act as an input port. For this application it must be set high since port 278H reads the input from the ADC.

TABLE III-2. Port Assignments

Port 278H - Input

IBM	25 PIN CONNECTOR	PIN ASSIGNMENT
D0	2	B1/B9 of ADC
D1	3	B2/B10 of ADC
D2	4	B3/B11 of ADC
D3	5	B4/B12 of ADC
D4	6	B5/POL of ADC
D5	7	B6/OR of ADC
D6	8	B7
D7	9	B8

Port 279H - Input

IBM	25 PIN CONNECTOR	PIN ASSIGNMENT
D3	15	Status of ADC

Port 27AH - Output

IBM	25 PIN CONNECTOR	PIN ASSIGNMENT
D0	1	Enable Latches
D1	14	LBEN* of ADC
D3	17	HBEN* of ADC
D5	set pin high through software	

Port 378H - Output

IBM	25 PIN CONNECTOR	PIN ASSIGNMENT
D0	2	*Latched to D4 of DAC
D1	3	*Latched to D5 of DAC
D2	4	*Latched to D6 of DAC
D3	5	*Latched to D7 of DAC
D4	6	*Latched to D8 of DAC
D5	7	*Latched to D9 of DAC
D6	8	*Latched to D10 of DAC
D7	9	*Latched to D11 of DAC

TABLE III-2. Port Assignments (Continued)

Port 37AH - Output

IBM	25 PIN CONNECTOR	PIN ASSIGNMENT
D0	1	*Latched to D4 of DAC
D1(-)	14	Latched to D5 of DAC
D2	16	*Latched to D6 of DAC
D3(-)	17	Latched to D7 of DAC

*Latched - signal is inverted before going to DAC

(-) - indicates an active-low signal

IV. Software Development

Now that the hardware circuit and interface to the computer have been designed and constructed, software is required to control the circuit, to run microbalance experiments, and to store the data from the experiments. This chapter shows the development of the software, including design and programming, which is needed to control the designed circuit of Chapter III.

Software Design.

There are two major functions of the software, zeroing the balance and weighing the sample. Although the two functions are discussed separately, both must perform the following activities: retrieve input from an addressable port, compare the input to a zero value, and output a value to an addressable port.

It is assumed that the microbalance responds to the output of the control circuit before the next sample is taken. This assumption is made since the changes in current to the microbalance solenoid are so small (nanoamps), the the reaction of the balance to the minute changes in the magnetic field of the solenoid is fast (at least faster than the ADC can perform conversions). However, if this assumption is incorrect, it is possible to change the sampling rate, using software, so that samples are taken only after the balance has reacted to the previous output.

Zeroing the Balance. Before a microbalance experiment begins, the balance must in the null position (i.e. the balance must not have an error signal). However, the tare weight and the sample rarely have the same mass. Thus the control circuit must "pre-balance" the sample and tare weight before the experiment runs. This pre-balancing is called zeroing the balance.

For this application, it is necessary for the mass of the sample to be equal to or greater than the mass of the tare weight. The reason for this is that a tare weight which is heavier than the sample would require a negative current through the solenoid for equilibrium, but the digital-to-analog converter (DAC) is configured to only output positive values. Thus, the assumption is made that the mass of the sample is greater than or equal in mass to the mass of the tare weight. Accordingly, it is also assumed that there will be no experiment performed on the balance where the sample loses mass.

The flow chart which represents the software for zeroing the balance is shown in Figure IV-1. The first thing that the program does is read input from the analog-to-digital converter (ADC). This input represents the error signal from the balance. If the error signal is zero, than the balance is in equilibrium and the balance is zeroed. When the balance is zeroed, the output to the DAC, given by DACOUT, is stored as DACZERO, which the weighing routine will refer to later. If the balance is not zeroed, there will be an error

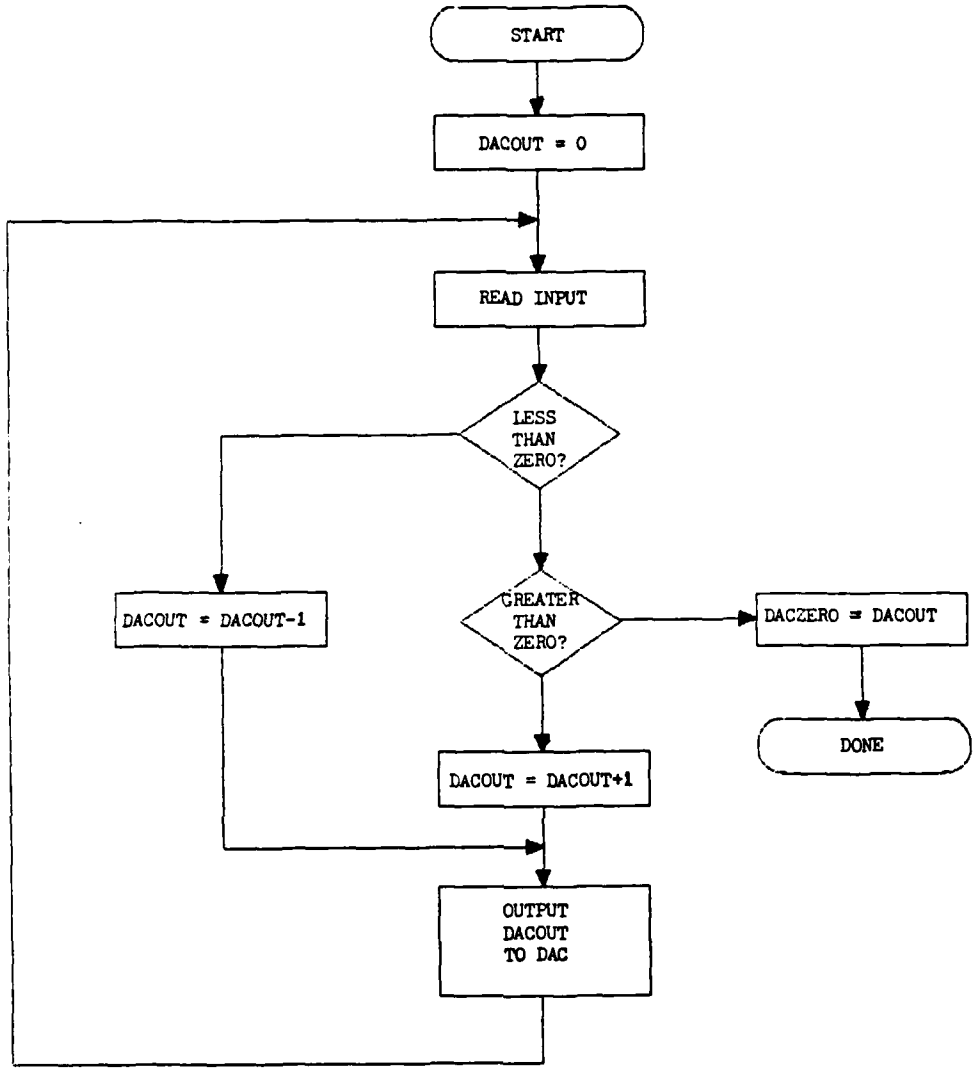


Figure IV-1. Flowchart for Zeroing the Balance

signal, and, consequently, an input from the ADC which is not equal to zero. In this case, DACOUT must be increased or decreased (DACOUT is decreased only if output to the solenoid has driven the balance over the null position), thus modifying the force on the tare side of the balance to force the

balance closer to equilibrium. Now another reading must be taken from the ADC to check if the balance is zeroed. If it is zeroed, the process will halt and store the value of DACOUT into DACZERO, or else it will keep checking and modifying DACOUT until the balance is zeroed. When the process is complete, the balance is zeroed. The value of DACZERO represents the current which the solenoid needs in order to hold the balance in equilibrium before the start of the balancing experiment.

In practice, the input from the ADC should be zero for several sampling periods to ensure that the balance is actually zeroed and not just passing through the null position at the time of sampling. In the case of this application, the balance must remain in the equilibrium position for ten consecutive sampling periods before the balance is considered to be zeroed. Also, if the input from the ADC is large (greater than 100H), the output to the DAC is incremented by ten instead of by one to zero the balance faster.

Weighing the Sample. The algorithm for the weighing routine is similar to the zeroing routine except that the input from the ADC is compared with the value stored in DACZERO rather than zero and selected data samples are stored in a memory location at the rate of one per second during the experiment and into a disk file after the experiment. Another difference between the two routines is that the zeroing routine is completed as soon as the balance is in

equilibrium, whereas the balance starts out in equilibrium at the beginning of the weighing routine and is back in equilibrium at the end of the routine. This presents a problem since the computer ends the weighing routine when the balance is in equilibrium. This means that the computer ends the weighing of the sample before the experiment actually starts. To correct this problem, the computer checks to make sure that the sample has gained mass and that the balance is in equilibrium before ending the routine. The flowchart for this weighing routine is shown in Figure IV-2.

At the start of the process, the output to the DAC, DACOUT, contains the value which is needed to zero the balance before the experiment has started. The time representing the beginning of the experiment is then retrieved from the computer. The time is used in the next step when the computer determines if one second has elapsed. If so, the value of DACOUT - DACZERO, which represents the amount of mass that the sample has gained, is stored in a memory location. The computer then reads the input from the ADC and proceeds through the same process used in zeroing the balance with slight modifications. The input from the ADC is compared with DACZERO. If less than DACZERO, then the output value, DACOUT, is decremented. If the input is greater than DACZERO, then DACOUT is incremented. In either of the above cases, the new value of DACOUT is sent to the DAC and the routine goes through the process again. However, if the input is equal to DACZERO, the balance is in equilibrium. In

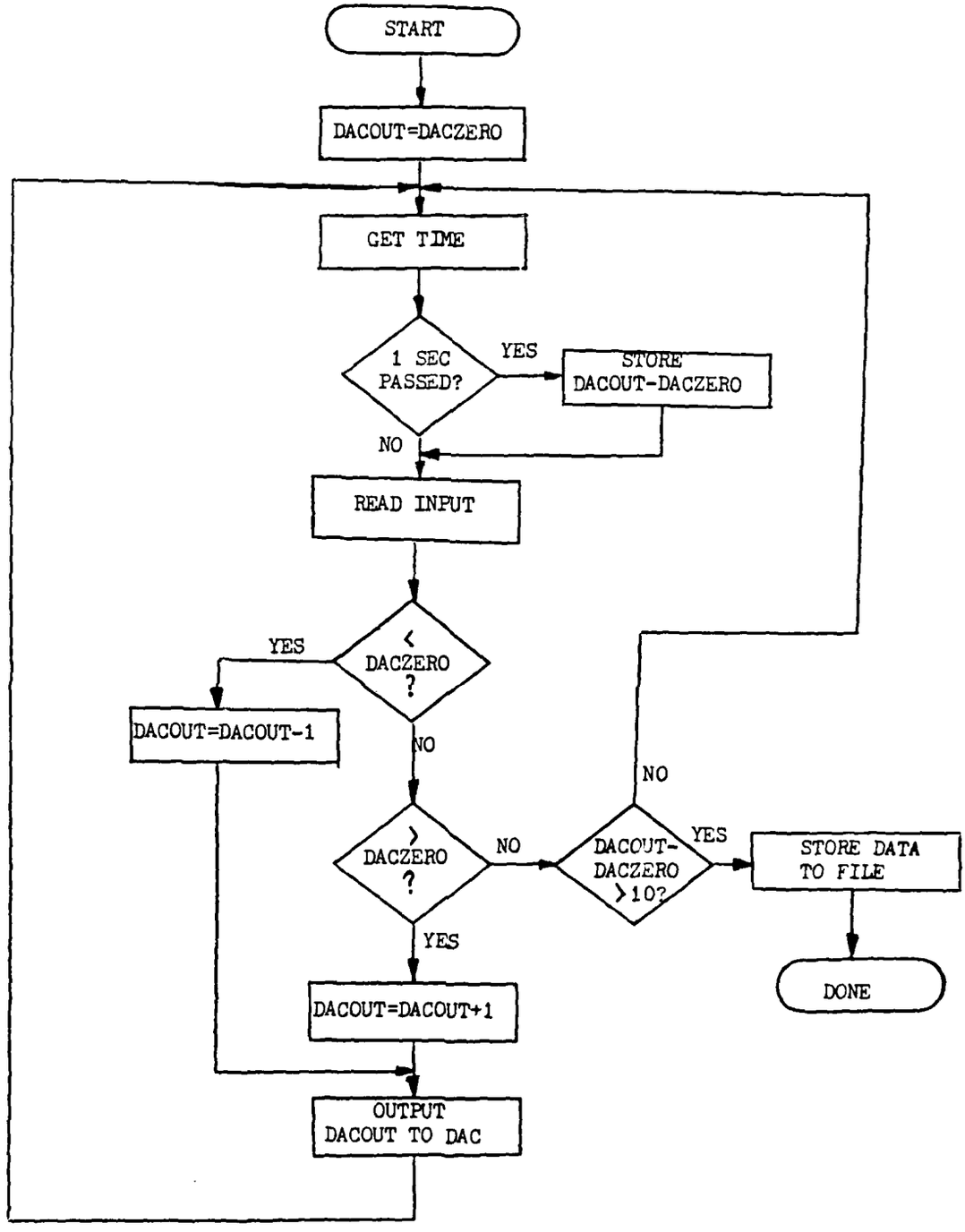


Figure IV-2. Flowchart for Weighing the Sample

this case, the computer must check to see if the sample has gained any mass by comparing the value of DACOUT with DACZERO. If the sample has not gained mass, then the computer must keep sampling and testing the input. If the sample has gained mass, then the experiment is over and the values of DACOUT that were stored in memory are written to a disk file.

A structure chart for the software program which includes zeroing the balance and the weighing the sample is shown in Figure IV-3. The program first initializes the variables, ports, etc. The zeroing routine is then called. After the balance is zeroed, the program prompts the user to adjust the necessary microbalance valves to start the experiment. When a keyboard entry has been made, the computer calls the weighing routine. This program continues until the end of the experiment. The data values are then stored in a disk file and the program is finished.

Software Programming.

In this section of the chapter, the actual programming of the software is discussed. The programming language used is the 8086/8088 assembly language. This language was chosen because of its speed and ability to manipulate I/O port bits. The speed is needed since the optimum sampling rate is the rate at which the ADC can convert the analog signal to digital form (7.5 per second). The manipulation of port data is also needed since the I/O ports are the only means

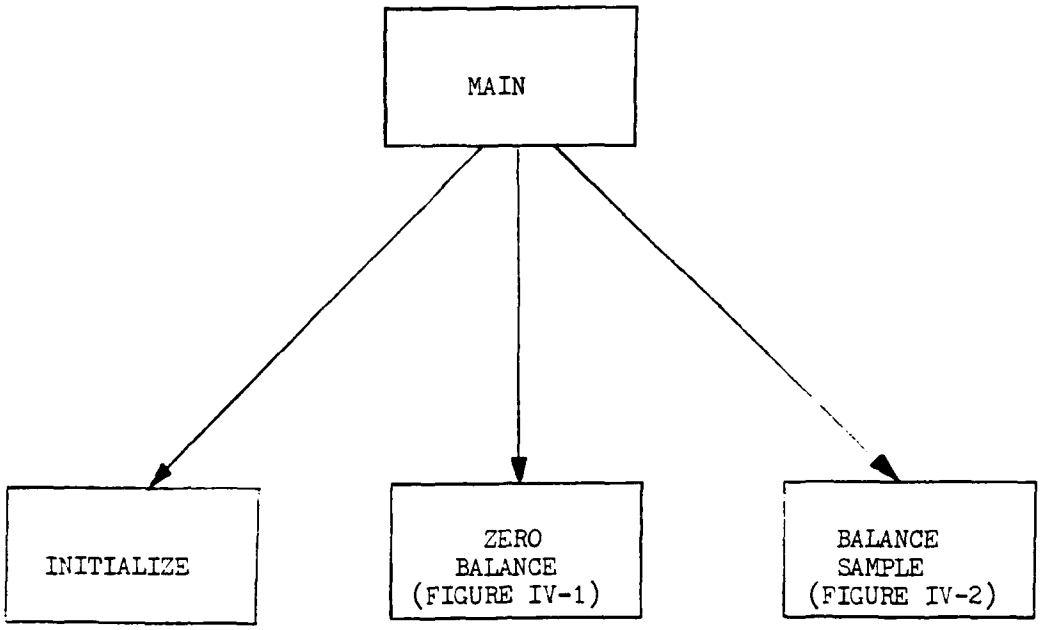


Figure IV-3. Structure chart of Main Routine to Run the Microbalance Experiment.

of communication the computer has with the hardware circuit. The software program that was written and is referred to by this section is given in Appendix E.

Since the program in Appendix E is in a modular format, the program is described by referring to its routines. The structure chart in Figure IV-3 represents the main program, called MAIN. The flowcharts in Figures IV-1 and IV-2 have been converted into primary routines called ZERO and BALANZ, respectively. An additional primary routine, INIT, has been added for initialization of the program. For ease in converting the flowcharts into programming, a few secondary subroutines are used. These subroutines are called INPRT, GETOPT, OUTPRT, DISPLY, CKDONE, STORE, and DISK. The sub-

routines and, subsequently, the primary routines are discussed during the remainder of this chapter.

INPRT This routine waits for the ADC status bit (bit 4 of port 279H) to go from a high to low transition before it accepts input from the ADC (see Chapter III for details). As soon as the status bit transition occurs, the computer enables the low order byte of the ADC using the control lines to LBEN* and HBEN*, bit 2 and bit 4 of port 279H, respectively. After reading the low order byte input from the ADC (bit 1 - bit 8 of port 2787H), the computer enables and reads the high order byte of the ADC (also, bit 1 - bit 8 of port 278H). If the sample is too heavy for the balance (i.e. out of range), the overrange bit of the ADC (bit 5 of the high order byte) is set high and the computer sounds a buzzer to notify the operator. Next, the computer checks the polarity bit (bit 6 of the high order byte). If this bit is low, the input from the ADC is a negative number and the computer takes the two's complement of the input to use in its computations. The input (or two's complement if negative) is then stored as a variable and the routine returns to the calling routine.

GETOPT This routine determines the value which is sent to the DAC. If the input from the ADC is zero, the balance is at the null position and the routine does not change the output value to the DAC. The output value is incremented if the input is greater than zero and decremented if less than

zero. If the magnitude of the input (absolute value) is greater than 100H, the output to the DAC is incremented or decremented by ten rather than by one (Note this can only happen during the zeroing of the balance since the balance never varies far from the null position during the experiment). The reason for incrementing or decrementing by ten is to force the balance to move quicker towards the null position when it is being zeroed. Since the DAC is configured to output only positive numbers, the output to the DAC does not decrement if it is equal to or less than zero. The routine then returns the output value to the calling program.

OUTPRT. This routine performs the actual output operation to the latches and DAC. Since the output ports are only 8 bits, the 12 bit output to the DAC must be accomplished in two operations. First, the upper eight bits are output to port 378H. Then, the lower four bits are output to port 37AH. These two ports are connected to the latches. After all 12 bits of data are sent to the latches, the latches are enabled so that the output is received by the DAC. This is done by setting the sixth bit of port 27AH active-high.

DISPLY. This routine converts the current hexadecimal value in the CX register to ASCII format and displays it on the console. First, the routine displays a negative sign (-) if the value in the CX register is negative. Then, the four most significant bits (bits 8-12) of the value in CX are converted to an ASCII digit and displayed. Next, the middle

four bits in the CX register are converted and displayed followed by the least significant four bits. The displaying of the bits is done using the video interrupt to the CPU (16:7.33-7.39).

CKDONE. The purpose of this routine is to determine whether the balance has been at the null position long enough to be considered balanced and not just passing through the null position. This routine checks to make sure the output to the DAC has not changed within the last ten sampling periods. If the DAC output has changed (balance still fluctuating), the routine clears a flag bit (CY flag) and returns to the calling routine. If the balance has remained at the null position for the last ten sampling periods, this routine sets the CY flag bit and returns to the calling routine. The calling routine then checks the CY flag to determine if the balance is at the null position.

STORE. The purpose of this routine is to store the value which is sent to the DAC into memory. Since the computer samples the balance 7.5 times per second and there is a limited amount of memory available for use as storage, this routine only stores the output value one time per second. This enables the experiment to run 7.5 times longer than if each sample were stored. The size of stack that this program reserves for storing the output values is 65,536 bytes. Each output value to the DAC is composed of two bytes. This means that 32768 (65,536/2) samples may be taken. Since one value

is stored every second, the experiment may last up to a maximum of 9.1 hours. If 9.1 hours is an insufficient amount of time to complete the experiment, the size of the stack may be varied with additional programming (not to exceed the size of the free user memory).

When this routine is called, the current time is obtained from the computer so that it can check to see if one second has passed since the last time data was stored. If one second has passed, the current output value minus the zero offset value is stored on a specified stack in memory. If one second has not passed, the subroutine returns to the calling routine. In this routine, a check is made to see if 32,768 values have been stored on the stack, and, if so, the routine calls the procedure to store all the output data onto a disk file and ends the experiment.

DISK. This routine creates a file, DATA.TXT, and writes all of the output data that was previously placed on a specified stack into DATA.TXT. This data is placed in the file in hexadecimal format instead of ASCII format so that disk space may be conserved. If there is an error (i.e. no room on disk or bad sector on disk), program ends and no data is saved.

INIT. This procedure performs three functions: initializes the ADC port to be an input port, initializes the DAC output to zero, and clears the console.

Recall from the computer interface discussion of Chapter III, D5 of port 27AH must be set high so that the

computer realizes that port 278H is an input port. This instruction is performed at the same time the DAC latches are enabled since the enable bit (D0) is of the same port. The latches are enabled so a value of zero may be sent to the DAC. Since the program outputs various data to the console, the instructions to clear the screen and to initialize it to the 80 X 25 mode are executed.

ZERO. This procedure is outlined by the flowchart in Figure IV-1. Using the flowchart as a guideline, the structure chart in Figure IV-4 was designed. The purpose of this routine is to zero the balance before the weighing of the sample begins. First, INPRT is called to retrieve input from the ADC. Next, the output value to the DAC is determined by GETOPT, and then sent to the DAC by OUTPRT. After the output value is displayed on the console using DISPLY, CKDONE is called to determine if the balance is zeroed. The routine continues to loop until CKDONE sets the carry flag, indicating that the balance is zeroed. At this point, the output value to the DAC is stored in memory location, ZRO_DAC, the routine prompts the user to start the experiment and then returns to the main program.

BALANZ. This routine controls the actual weighing of the sample and is outlined in the flowchart of Figure IV-2. To assist in the programming, a structure chart, shown in Figure IV-5, was constructed. First, this routine retrieves the input from the ADC using INPRT. The STORE routine is

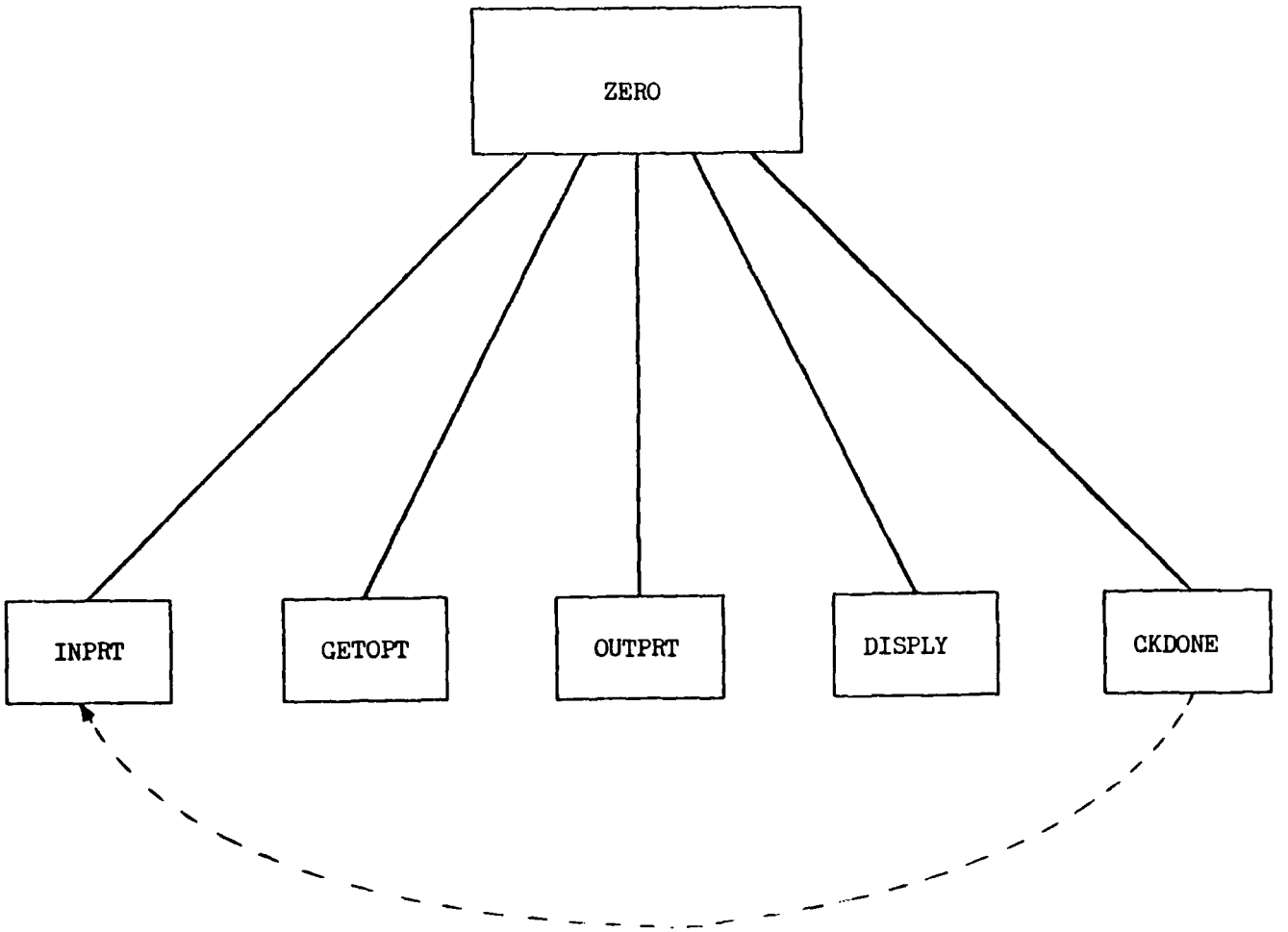


Figure IV-4. Structure Chart for Zeroing the Balance

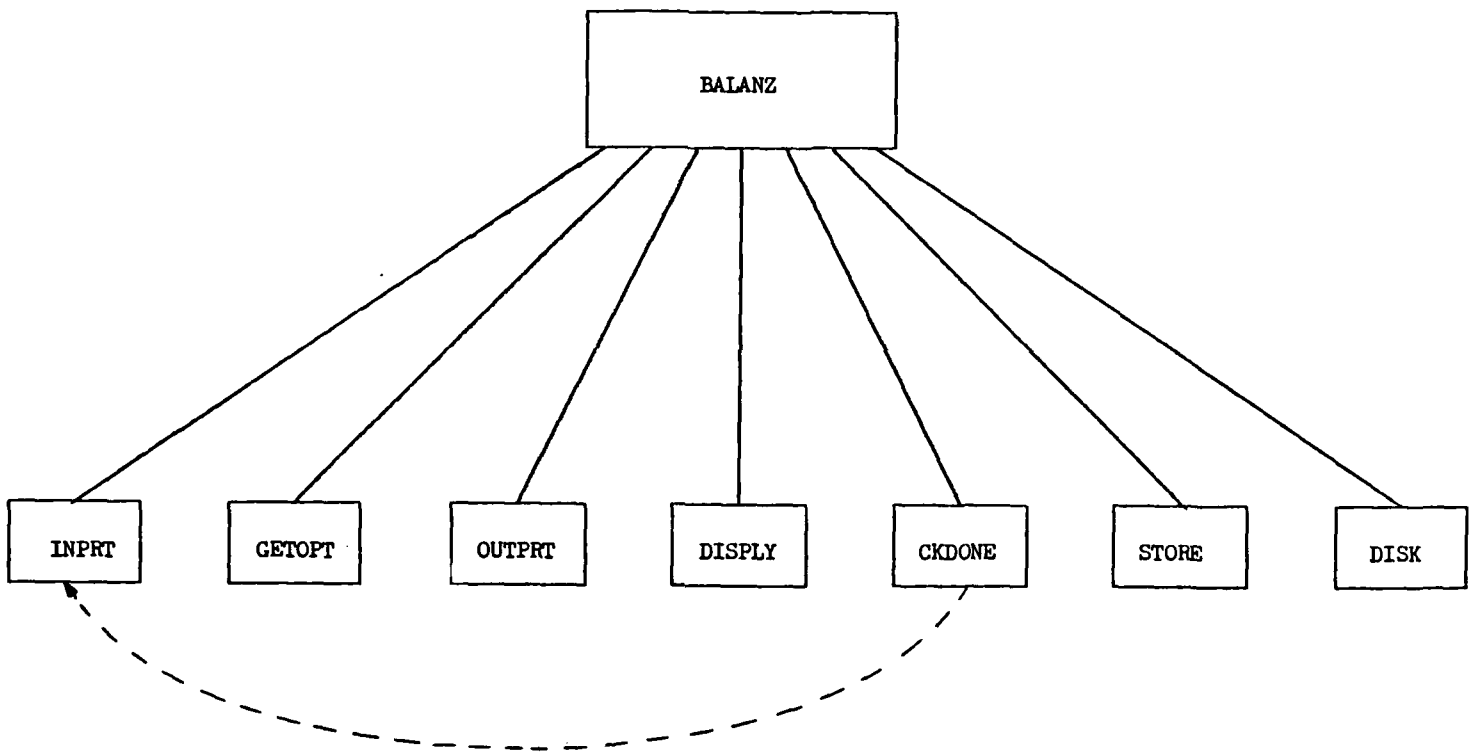


Figure IV-5. Structure Chart for Weighing the Sample

then called so that the current output to the DAC minus the zero offset, ZRO_DAC, is stored on a specified stack in memory if one second has elapsed. BALANZ then determines the output value to the DAC using GETOPT and then outputs that value to the DAC using OUTPRT. Next, the output value minus the zero offset is displayed on the console using DISPLY. A check is made to see if the sample on the balance has gained any mass. If a mass gain is detected, the routine continues driving the balance to the equilibrium position. If a mass gain is not detected, the routine calls CKDONE to see if the balance has been at the equilibrium position for the last ten sampling periods. If CKDONE does not set the CY flag, the balance is not in equilibrium and the routine continues to drive the balance to equilibrium. If CKDONE sets the CY flag, indicating that the experiment is done, then BALANZ calls DISK to store all of the output data on a disk file and then returns to the main program.

MAIN. This program, whose structure chart is shown in Figure IV-3, calls the primary routines listed above to perform the function of balancing the sample on the micro-balance. The first task performed in the program is to initialize code segment, data segment, and extra segment. Both the code segment and the data segment point to the segment called CODE, where the main, primary, and secondary routines exist. The extra segment points to DATA, the segment where the output data to the DAC is stored. The

program then calls INIT to initialize the data values and the ports. ZERO is called to zero the balance. And finally, BALANZ is called to control the actual weighing of the sample on the microbalance and store the data to a disk file.

The software in this chapter is a minimum of what is actually needed to control the microbalance. The software controls the zeroing of the balance and the balancing of a sample by retreiving input from a port, comparing the input to a zero value, and then outputting a value to a second port. Before an experiment begins, the user must be certain that 1) the mass of the sample is greater than or equal to the mass of the counterweight, 2) the sample will not lose mass during the experiment, and 3) there is enough room on the disk to store the data (at least 65,536 bytes). If any of the above conditions are not met, the data, if any, is erroneous. At the end of the experiment, the data values are stored in a file named DATA.TXT. These values are stored in hexadecimal format in order to use a minimal amount of disk space.

V. Testing

This chapter describes the testing performed on the digital system described in Chapters III and IV. Testing is done in two phases, hardware testing and software testing. The hardware testing is performed in order to verify that the hardware components of the digital system were connected correctly. The hardware testing is done concurrently with the testing of the computer interface making it possible to test the hardware and interface using a few software routines. After the correct operation of the hardware is verified, the software and, hence, the overall system is tested. This test is performed by constructing a circuit which simulates the response of the microbalance.

Hardware Testing.

The primary hardware components which required testing were the ADC, DAC, latches and computer interface. The testing of the ADC computer interface is presented simultaneously with the testing of the ADC. A similar discussion is given for the DAC computer interface.

ADC. The supplementary materials needed to test the ADC and its interface with the computer are a variable DC voltage source with a voltage display, and a software program. The DC source is connected to the ADC as shown in Figure V-1. The function of the software program, listed in Appendix F, is to read data from the ADC and output the data to the

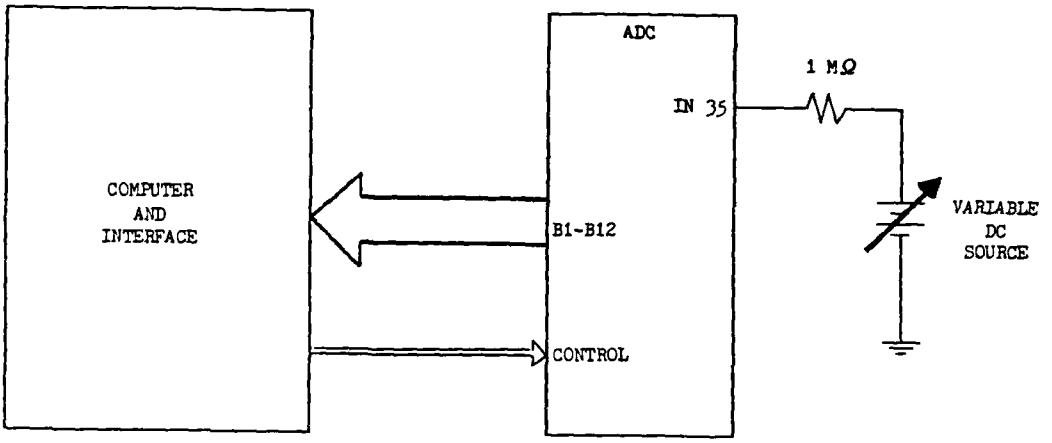


Figure V-1. Circuit for Testing the ADC

console. As the input voltage to the ADC is varied, the digital output also varies, as shown in Table V-1. The test is performed to assure that changes in the input voltage to the ADC result in changes to the ADC's digital output, accordingly. This test is positive since an increasing input voltage gives an increasing digital output. Also, a test is performed to assure that an input voltage of zero volts yields a zero digital output, and an input voltage of three volts yields the maximum digital output, 0FFFH. As shown in Table V-1, this test also proves positive. To determine the accuracy of the ADC (i.e. the minimum voltage change needed to vary the output), the voltage input value is divided by its corresponding digital output value. This calculation results in an average accuracy of 0.74 millivolts. Since

there is an instrumentation amplifier in the actual digital system with a gain of 300, the ADC is able to detect a change in the error signal from the balance with an accuracy of 0.74/300 = 2.45 microvolts. However, this resulting accuracy is theoretical since it was not possible to test an accuracy this small without a voltage source capable of delivering a precise 2.45 microvolt increment.

To verify the proper functioning of the computer interface, it is necessary to show that the data received by the computer is the same as the output of the ADC. This test is accomplished by comparing the output of the ADC (using the breadboard's built-in logic probe) with the console display. The result is that the computer received the correct ADC output, and thus, the interface between the ADC and the computer functions properly.

TABLE V-1. ADC Testing Results

Analog Input Volts	Digital Output Hexadecimal
0	000H
0.1	089H
0.2	110H
0.3	198H
0.4	220H
0.5	2A8H
0.6	330H
0.7	3B7H
0.8	43FH
0.9	4C7H
1.0	550H
1.5	7FAH
2.0	AA6H
2.5	D50H
3.0	FFFH

DAC and Latches. The additional materials used to test the DAC and latches are a Simpson 460 voltmeter and a software program. The voltmeter is placed across the DAC output as shown in Figure V-2. The function of the software program, shown in Appendix F, is to increment the output to the DAC every time a keyboard entry is made. The results of this test are shown in Table V-2. This test shows that the DAC has an average incremental output of approximately 1.16 millivolts. Since there is a voltage divider in the designed digital system which reduces the output of the DAC by a factor of 1500.0/13.0 = 115, the incremental output that is seen by the resistor and solenoid is (1.16 mV)/115.4 = 10 microvolts. To test the latches and their interface to the computer, the logic level of the input pins on the DAC are

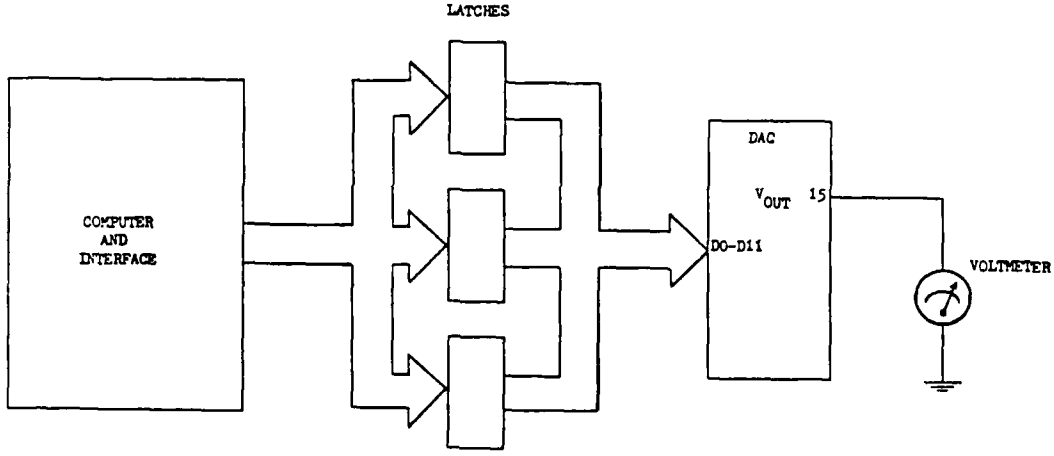


Figure V-2. Test Circuit for the DAC

manually checked as keyboard entries are made. As expected, the input to the DAC is incremented as keyboard entries are made. This test shows that the latches and the computer interface function properly.

TABLE V-2. DAC Testing Results

Number of Keyboard Entries	Analog Output mV	Average Step Difference, mV
0	0	-
1	1.15	1.150
2	2.30	1.150
4	4.60	1.150
8	9.20	1.150
16	18.60	1.163
32	37.15	1.161
64	74.55	1.165
128	149.10	1.165
256	298.00	1.164
512	596.00	1.164
1024	1192.00	1.164
2048	2375.00	1.160
4095	4725.00	1.154

Software Testing.

The testing of the software is performed concurrently with the overall system test. The additional equipment used to test the system is a Simpson 460 voltmeter and a Datell calibration source (variable DC voltage source) which outputs millivolts without a significant noise level.

In order to test the designed digital system, a feedback loop is connected between the input and output of the system, shown in Figure V-3. Recall, the assumption is made that the changes in current to the microbalance solenoid are so small

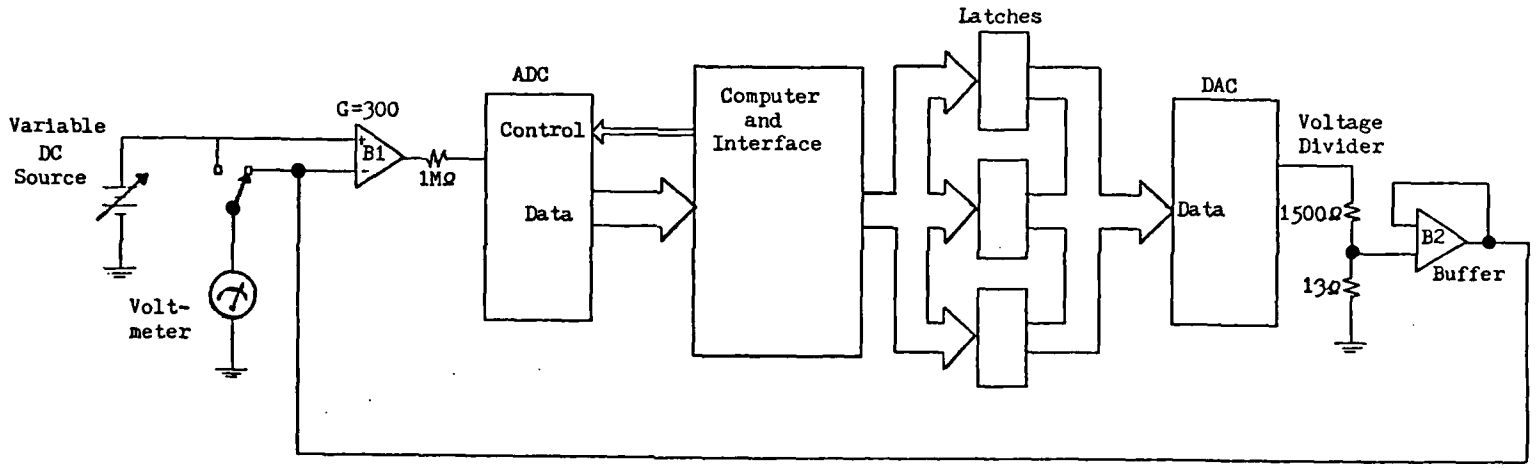


Figure V-3. Circuit for Testing Software

that the balance's response to the solenoid settles before there is another change in the current. This means that the microbalance settles before the computer receives the next sample from the ADC. Therefore, this feedback loop simulates the response of the microbalance to the digital system. The DC voltage source is used as the external force on the microbalance (i.e. the mass of the sample). The simulation of the mass change of the sample is performed when the voltage input is increased. The output voltage of the digital system is fed back into the instrumentation op amp (see Figure V-3). Here, the difference between the output voltage and the external voltage is received by the ADC. An equilibrium position is simulated when the input to the ADC is zero. This means that the output of the system is equal to the external voltage.

The first part of the test is to set the external voltage source to 5 mV. This simulates the mass difference between the sample and the tare and tests out the ability of the system to zero the balance. The program is run and the output of the digital system is driven from 0 to 5 mV $\pm$ 20 μV. The program then prompts the user to start the next process, which is to balance the sample. An example of a mass change during an actual oxidation study is given in Figure V-4 (1:317). To simulate the case presented above, the input voltage is increased at a rate of approximately one quarter of a millivolt per second until 9 millivolts was reached. At this point in time, the rate at which the

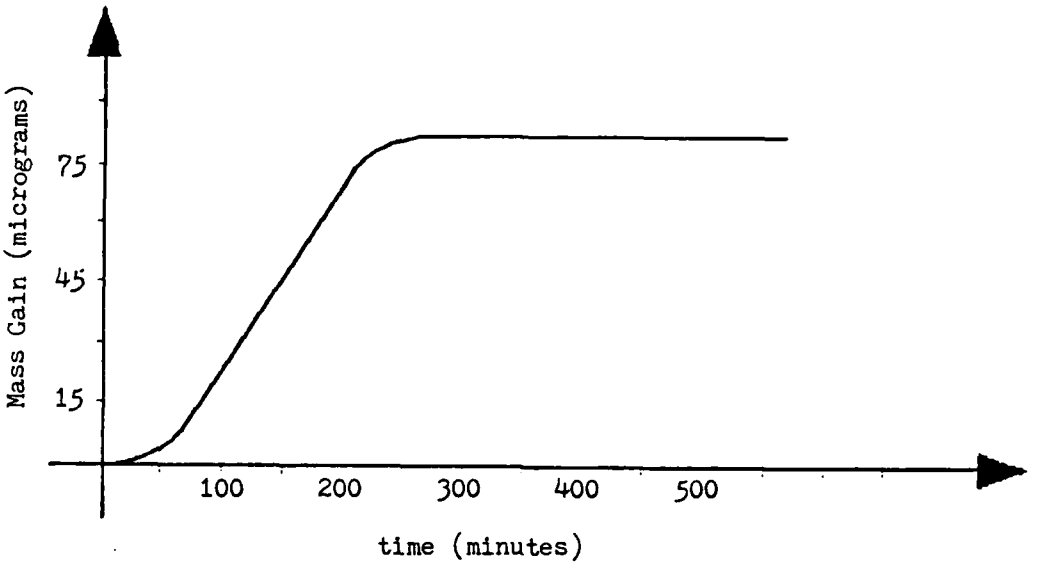


Figure V-4. Example of a Mass Gain in an Oxidation Experiment (1:317)

external voltage input changes is decreased until there is a constant external input voltage. The difference between the final output voltage and the initial output voltage is representative of the mass gain of the sample (recall the compensating force on the balance is proportional to the mass gain). As the above simulation is being performed, the output of the digital system tracks the increase of the voltage source and is always within ± 20 μV of the voltage source. When the voltage source stopped increasing (at 9 mV), the program halts the experiment and stores the data on the disk in a file named DATA.TXT. Inspection of the contents of DATA.TXT reveals that the DAC output is stored as 3-digit hexadecimal values starting at the value 000H and

increasing to the value 217H, representing a gain from 0 mV to 4 mV.

Also run on the system are error checking test in case of some unknown condition occurring. An initial negative input to the system instructs the user to either decrement the tare switch or place a lighter sample on the balance (i.e. the sample is lighter than the counterweight). If the output of the DAC is incremented higher than 0FFFH or decremented lower than 000H, a buzzer sounds to warn the operator of a possible problem (i.e. sample gains more mass than circuit can compensate or the sample loses mass).

To determine why the desired system output is in error up to 20 microvolts, the test equipment was examined. Results show that the ground throughout the circuit varied by as much as 6.3 mV. That is, there was a potential difference of 6.3 mV between the ground on one end of the circuit and on the other end of the circuit. It is extremely likely that this problem was contributed to the error in the system's output. However, the problem is due to the resistance of the Elite 3 breadboard, and could not be eliminated without reconstruction of the circuit on a pc-board or wirewrapped board.

VI. Results and Recommendations

Results.

In Chapter II, an analysis of the analog circuit presently used to control the microbalance and an indepth description of a portion of the circuit, the derivative compensator, are presented. This analysis shows that the overall function of the circuit is to 1) receive a very small error signal from the balance infrared detectors, 2) record the voltage and time of data continuously, and 3) output a very small current to compensate for the input voltage (i.e. drive the input voltage to zero).

The hardware circuit designed in Chapter III is capable of detecting changes in the error signal from the microbalance of 2.45 microvolts, and applying changes to the compensating signal in increments of 10 microvolts. The accuracy of the digital system (i.e. how closely the output follows the input) is designed to be within the output step increment of 10 microvolts. However, the noise produced by the ground potential of the test equipment decreases the accuracy by 20 microvolts.

The software developed in Chapter IV controls the operation of the balance via the hardware circuit. The software, listed in Appendix E, was written in 8088 assembly language due to speed requirements and the need to manipulate I/O port bits. After reading an error signal from the balance, the software provides the feedback needed to force the micro-

balance to its equilibrium position by either incrementing or decrementing the output of the digital control system (i.e. the software decrements the output if there is a negative error signal and increments the output if there is a positive error signal). If the balance is in its equilibrium position, the error signal is zero and, the software makes no changes to the output of the system since no additional feedback is needed. In addition to providing the proper feedback to the balance, the software stores the data points into memory and then onto a disk file.

The digital system performs in a manner similar to the analog circuit described above. It, also, reads a small error signal input (less than 10 millivolts) from the balance, records the data and time of sample, and outputs a small current which forces the balance to its equilibrium position. In a simulated microbalance experiment, the digital system initially zeroed the balance, weighed the sample, and then stored the resulting data to a file named DATA.TXT, in hexadecimal format.

The digital feedback system designed in this thesis is capable of replacing the analog control circuit described in Chapter II. There are, however, some assumptions made for correct operation of the designed system:

1. The maximum error signal from the balance infrared detectors is ten millivolts.
2. The balance response to the output of the digital system is complete before the next sample is taken.

3. There is enough room on the disk to store data points (at least 65,536 bytes of free space).

4. The sample is always heavier than the counterweight.

5. The sample does not lose mass during the experiment.

Recommendations.

The following recommendations are made for further enhancement of the digital control system designed in this thesis:

1. Construct the circuit on a printed circuit board in order to alleviate the problem of a floating ground.

2. Replace the 12-bit DAC with a high linearity 16 bit DAC so that both higher accuracy and linearity are obtained.

3. Configure the DAC to output both negative and positive voltages so that samples may lose as well as gain weight. Software changes must be included to accommodate this DAC modification.

4. Increase the size of the data stack in the software routine to 131,072 bytes so that a greater number of data points may be stored and also to enable the experiment to run twice as long.

5. Prompt user to place another disk into computer if the current disk does not have enough space to store all of data. This prevents having to perform the experiment again if the data does not fit on the disk.

6. Create a software routine which plots out the data stored in DATA.TXT to allow easier analysis of data.

APPENDIX A: MICROBALANCE PARTS LEGEND

The following is a parts legelnd of the microbalance shown in Figure III-1 (11:712).

(A) Balance housing

(B) Quadrupole mass filter

(C) Turbomolecular pump

(D) Line to gas handling system

(E) Leveling platform

(F) Support table

(G) Transite top

(H) Hangdown tubes

(I) Alignment bolts

(J) Solenoid assembly

(K) Solenoid support flange

(L) Windows

(M) "Y" fitting with port

(N) Motor-driven linear motion feedthrough

(O) Sample loading fitting

(P) Quick access flange

(Q) Four-way cross

(R) Six-way cross with push-pull feedthrough

(S) Capacitance manometer gauge tube

(T) Tublulation for Pumping Reference Side of (S)

(U) Straight through valves

(V) "Tee" fitting

(W) Flexible bellows

(X) Manifold

(Y) Vacuum bakeout oven

(Z) Dry box

(AA) Uncoupling point for sample suspension fibers

(BB) Bellows

(CC) Location for changing sample and then taring after resuspending sample

(DD) Access doors to vacuum bakeout oven

(EE) Vacuum or backfilling line

(FF) Tube Furnaces

APPENDIX B: DERIVATIVE COMPENSATOR CIRCUIT

The derivative compensator circuit, shown in Figure B-1, for the analog system described in Chapter II does not have an obvious transfer function. In order to determine the function of this configuration, as used in the system of Chapter II, it is necessary to derive the transfer function.

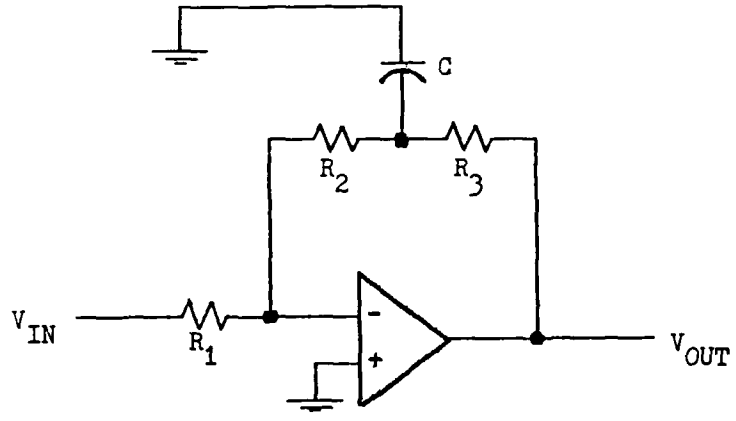


Figure B-1 Derivative Compensator Circuit.

To get the transfer fucntion of the derivative compensator circuit shown in Figure B-1, the output voltage, V_{OUT}, must relate to the input voltage, V_{IN}. To assist in doing this, the intermediate voltages, V_1 and V_2, must be used.

Using Kirchoff's voltage laws, a relationship between V_{IN}, V_1, and V_2 is given as

$$\frac{V_1 - V_{IN}}{R_1} = \frac{V_2 - V_1}{R_2} \qquad \text{(B-1)}$$

V_{OUT} can be related to V_1 and V_2 using the same law:

$$\frac{V_2 - V_1}{R_2} + \frac{V_2}{R_3} + V_2(jwC) - \frac{V_{OUT}}{R_3} = 0 \qquad \text{(B-2)}$$

Since the positive terminal of the op amp is grounded, $V_1 = 0$, and equation (B-1) becomes

$$V_2 = \frac{-R_2 \cdot V_{IN}}{R_1} \qquad \text{(B-3)}$$

and rearranging equation (B-2), we get

$$\frac{V_{OUT}}{V_2} = \frac{R_3}{R_2} + 1 + R_3(jwC) \qquad \text{(B-4)}$$

By substituting equation (B-3) into (B-4) and rearranging the terms, the resulting transfer function is

$$\frac{V_{OUT}}{V_{IN}} = -\frac{R_2 + R_3}{R_1} \cdot [\, 1 + R_2 || R_3(jwC)] \qquad \text{(B-5)}$$

This transfer function shows that a signal entering the derivative compensator circuit will have a DC gain of

$$-\frac{R_2 + R_3}{R_1} ,$$

and its phase will be shifted by

$$[\, 1 + R_2 || R_3(jwC)].$$

APPENDIX C: PIN DESCRIPTION OF THE ICL7109 ADC

The following is a list of the pin assignments and function descriptions of each pin on the analog-to-digital converter, ICL7109, of the circuit described in Chapter III (7:4-44).

PIN	SYMBOL	DESCRIPTION
1	GND	Digital Ground, 0 Volts, Ground return for all digital logic.
2	STATUS	Output HI during integrate and deintegrate until data is latched. Output LOW when analog section is in Auto-Zero configuration.
3	POL	Polarity - HI for positive input.
4	OR	Overrange - HI if overranged.
5	B12	Data Bit 12 (most siginificant bit).
6	B11	Data Bit 11.
7	B10	Data Bit 10.
8	B9	Data Bit 9.
9	B8	Data Bit 8.
10	B7	Data Bit 7.
11	B6	Data Bit 6.
12	B5	Data Bit 5.
13	B4	Data Bit 4.
14	B3	Data Bit 3.
15	B2	Data Bit 2.
16	B1	Data Bit 1 (least sigificant bit).
17	TEST	Input HI for normal operation. Input LOW forces all outputs high.

Data bits all have a tri-state output and are active high.

PIN	SYMBOL	DESCRIPTION
18	LBEN*	Low-byte enable. Taking this pin LOW when pins 20 and 21 set LOW, activates the low order byte, B1-B8. Taking this pin HI, disables the output of B1-B8 into a high impedence state.
19	HBEN*	High-byte enable. With pins 20 and 21 set LOW, taking this pin LOW activates the high order byte outputs B9-B12, POL, and OR. Taking this pin HI, disables the high order byte outputs B9-B12, POL and OR.
20	CE/LOAD*	Chip enable. With pin 21 LOW, this pin serves as a master output enable for pins 3 - 16. When this pin is set HI, all outputs are diabled.
21	MODE	Input LOW - Direct output mode where CE/LOAD*, HBEN*, and LBEN* act as inputs directly controlling byte outputs. Input HI - Used for handshaking mode where CE/LOAD*, HBEN*, and LBEN* acts as outputs.
22	OSC IN	Oscillator input.
23	OSC OUT	Oscillator output.
24	OSC SEL	Oscillator select - Input HI configures OSC IN, OSC OUT, BUF OSC OUT as RC oscillator - clock will be same phase and duty cycle as BUF OSC OUT. - Input LOW configures OSC IN, OSC OUT for crystal oscillator - clock frequency will be 1/58 of frequency at BUF OSC OUT.
25	BUF OSC OUT	Buffered oscillator output.

PIN	SYMBOL	DESCRIPTION
26	RUN/HOLD*	Input HI - Conversions continuously performed every 8192 clock pulses. Input LOW - Conversion in progress is completed, ADC will stop in Auto-Zero seven counts before integrate.
27	SEND	Input - Used in handshaking mode to indicate ability of external devide to accept data.
28	V-	Analog negative supply - Nominally -5 Volts with respect to GND.
29	REF OUT	Reference voltage output - Nominally 2.8 Volts down from V+.
30	BUFFER	Buffer amplified output.
31	AUTO-ZERO	Auto-Zero node - Inside foil of C_{AZ}.
32	INTEGRATOR	Integrator output - Outside foil of CINT.
33	COMMON	Analog common - System is Auto-Zeroed to COMMON.
34	INPUT LO	Differential input low side.
35	INPUT HI	Differential input high side.
36	REF IN+	Differential reference input positive.
37	REF CAP+	Reference capacitor postive.
38	REF CAP-	Reference capacitor negative
39	REF IN-	Differential reference input negative.
40	V+	Positive supply voltage - Nominally +5 Volts with respect to GND.

APPENDIX D: MODIFICATION OF APPARAT COMBO CARD II

When originally purchased, the Apparat Combo II card contains a parallel printer port which is configured for "output only" of the 8 data bits of its memory addressed port 278H. By physically modifying the card, the port can be made to be bidirectional. That is, the port can receive input as well as output. This modification is done using the following procedure:

1) Locate points D, E, and F on the Combo II card. These points are on the top right hand section of the card and are represented in Figure D-1a.

2) Cut the trace between points E and F.

3) Insert a jumper between points D and E as in Figure D-1b [13:7].

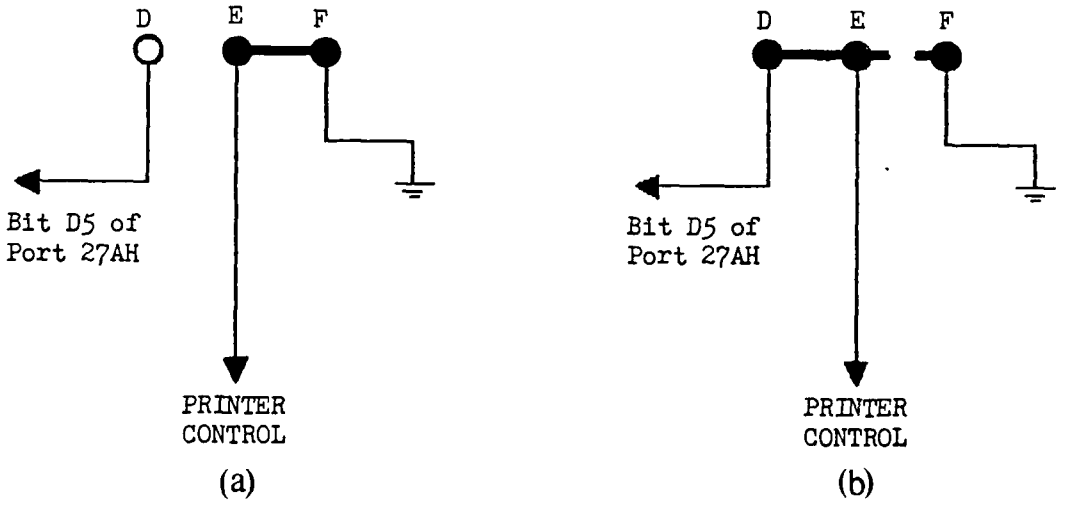


Figure D-1. Points D, E, and F on the Combo II card (a) before and (b) after modification.

Before the modification, point E, the trace to the printer control, is grounded. This is the equivalent of a logic '0' on the printer control which tells the Combo II card that port 278H will only be used to output data. For normal printer use this would be the case. In order to use port 278H as an input port, the printer control must receive a logic '1'. Before any modification of the card, inputting would be impossible since the printer control is physically tied to ground, or logic '0'. After modification, point E on the card was no longer tied to ground, but was instead connected to bit D5 of port 27AH. This allows the user to decide whether port 278H will be used as an in put or an output port. To use as an input port, the user must set bit D5 of port 27AH to logic '1', and for use as an output port, bit D5 of port 27AH must be set to logic '0'.

APPENDIX E: SOFTWARE TO OPERATE MICROBALANCE CONTROL CIRCUIT

```
CODE SEGMENT
;*****************************************************************
;
;  PROGRAM NAME:  BALANCE
;  VERSION:  1.0
;  SYSTEM:   MSDOS (Zenith 151)
;  LANGUAGE: 8088 Assembler (MASM)
;  AUTHOR:   2LT CHRIS BOLAN
;  DATE:     13 NOVEMBER 1985
;
;  This program controls is part of a digital system which
;  controls the balancing activity on an Ultragravimetric
;  Pivotal Microbalance.  The program first zeroes the
;  balance, and then balances the sample.
;
;*****************************************************************
;

MAIN    PROC    FAR
    ASSUME   CS:CODE,ES:DATA,DS:CODE


    MOV  AX,DATA
    MOV  ES,AX
    MOV  AX,CODE
    MOV  DS,AX
    CALL INIT           ; INITIALIZE PORTS AND REGISTERS
    CALL ZERO           ; ZERO THE BALANCE
    CALL BALANZ         ; START THE BALANCING ROUTINE
    JMP  END            ; RETURN TO OPERATING SYSTEM

MAIN    ENDP
```

```
;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  INIT
;  FUNCTION:      Initializes I/O ports, console screen, and
;                 output to DAC.
;  INPUT:         none
;  OUTPUT:        BX, the register which contains the output
;                 to the DAC is 0
;  REGISTERS:     none are preserved
;  MODULES USED:  none
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*****************************************************************
;
INIT  PROC  NEAR

    MOV  DX,27AH          ; INITIALIZE PORT 278H TO BE AN
    MOV  AL,00100000B     ; INPUT PORT BY SETTING D5 OF 27AH
    OUT  DX,AL            ;
    MOV  BX,0000H         ; INITIALIZE OUTPUT PORT TO ZERO
    CALL OUTPRT

    MOV  DX,3D8H          ; SCREEN IS 80 X 25 TEXT MODE
    MOV  AL,29H           ;
    OUT  DX,AL            ;

    INC  DX               ; MAKE BORDER BLUE
    MOV  AL,09            ;
    OUT  DX,AL            ;

    MOV  AX,0600H         ; CLEAR SCREEN
    MOV  BH,7             ; USE BLANKS TO CLEAR
    MOV  CX,0000H         ; UPPER ROW, COL OF SCREEN
    MOV  DX,184FH         ; LOWER ROW, COL OF SCREEN
    INT  10H              ;

    MOV  BX,0000H
    CALL PROMPT
    RET
INIT    ENDP
```

```
;
;*************************************************************
;
;  SUBROUTINE ENTRY POINT:  PROMPT
;  FUNCTION:      prompts user to hit any key on keyboard to
;                 continue
;  INPUT:         none
;  OUTPUT:        message to console
;  REGISTERS:     none are preserved
;  MODULES USED:  none
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*************************************************************
;
PROMPT  PROC  NEAR
    MOV   DX,OFFSET PRMSG     ; POINT TO PROMPT MESSAGE
    MOV   AH,09               ; DOS DISPLAY FUNCTION
    INT   21H                 ; DOS FUNCTION CALL
    XOR   AH,AH
    INT   16H
    MOV   DX,OFFSET BLANK     ; POINT TO BLANKS TO ERASE
                              ; MESSAGE
    MOV   AH,09               ; DOS DISPLAY FUNCTION
    INT   21H                 ; DOS FUNCTION CALL
    RET
PROMPT   ENDP
```

```
;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  ZERO
;  FUNCTION:      zeroes the balance
;  INPUT:         none
;  OUTPUT:        none
;  REGISTERS:     none are preserved
;  MODULES USED: CHKDAC, INPRT, GETOPT, PROMPT, DISPLY,
;                 CKDONE
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*****************************************************************
;
ZERO  PROC   NEAR

    CALL CHKDAC                 ; MAKE SURE DAC STARTS OFF
                                  ; POSITIVE
 CONT1:
    CALL INPRT                  ; GET A/D INPUT
    CALL GETOPT                 ; GET DAC OUTPUT
    CALL OUTPRT                 ; OUTPUT TO DAC
    MOV  DX,0D23H               ; POSITION OF INPUT DISPLAY
    CALL DISPLY                 ; DISPLAY INPUT
    CALL CKDONE                 ; CHECK TO SEE IF DONE ZEROING
    JNC  NOZERO
    MOV  WORD PTR ZRO_DAC,BX        ; STORE DAC VALUE TO ZERO
                                    ; BALANCE
    CALL PROMPT                 ; CONTINUE
    RET
NOZERO:
    MOV  CX,BX
    MOV  DX,0B23H               ; POSITION OF OUTPUT DISPLAY
    CALL DISPLY                 ; DISPLAY OUTPUT
    MOV  AH,01                  ; CHECK FOR KEYBOARD INPUT
    INT  16H                    ; IF SO, CHECK TO SEE IF INPUT
                                ; IS A ^C
    JZ   CONT1                  ; ELSE KEEP ZEROING BALANCE
    CALL CTRLC
    JMP  CONT1
ZERO    ENDP
```

```
;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  BALANZ
;  FUNCTION:      places balance in equilibrium
;  INPUT:         none
;  OUTPUT:        none
;  REGISTERS:     none are preserved
;  MODULES USED:  INPRT, STORE, GETOPT, OUTPRT, DISPLY,
;                 CKDONE, DISK
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*****************************************************************
;
BALANZ   PROC  NEAR

    MOV  AH,2CH              ; GET CURRENT TIME FOR START OF
                             ; PROCESS
    INT  21H
    MOV  BYTE PTR STO_CNT,DH

CONT2:
    CALL INPRT               ; GET A/D INPUT
    CALL STORE
    CALL GETOPT              ; GET DAC OUTPUT
    CALL OUTPRT              ; OUTPUT TO DAC
    MOV  DX,0D23H            ; POSITION OF INPUT DISPLAY
    CALL DISPLY              ; DISPLAY INPUT
;
;  CHECK TO MAKE SURE THAT BALANCE HAS STARTED MOTION
;
    MOV  AX,BX
    SUB  AX,WORD PTR ZRO_DAC
    CMP  AX,20
    JL   NOTBAL              ; IF BALANCE IS IN MOTION,
    CALL CKDONE              ; CHECK TO SEE IF WE'RE DONE

    JNC  NOTBAL
    CALL PROMPT              ; CONTINUE
    CALL DISK                ; WRITE OUTPUT TO A DISK FILE
    RET
NOTBAL:
    MOV  CX,BX
    MOV  DX,0B23H            ; POSITION OF OUTPUT DISPLAY
    CALL DISPLY              ; DISPLAY OUTPUT

    MOV  AH,01               ; CHECK FOR KEYBOARD INPUT
    INT  16H                 ; IF SO, CHECK TO SEE IF INPUT
                             ;  IS A ^C
    JZ   CONT2               ; ELSE KEEP BALANCING
    CALL CTRLC
    JMP  CONT2
BALANZ   ENDP
```

```
;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  CKDONE
;  FUNCTION:      checks to see if balance is in equilibrium
;  INPUT:         CX :  input from ADC
;                 BX :  output to DAC
;  OUTPUT:        CY FLAG
;  REGISTERS:     all are preserved
;  MODULES USED:  none
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*****************************************************************
;
CKDONE   PROC  NEAR
    CMP  CX,0000H              ; CHECK IF INPUT IS ZERO
    JNZ  NOTDON                ; IF NOT, JUMP TO NOT DONE
    CMP  BX,LASTDAC            ; ELSE COMPARE CURRENT OUTPUT
                               ; TO LAST
    JNZ  NOTDON                ; OUTPUT. IF NOT EQUAL, YOU'RE
                               ; NOT DONE
    DEC  BYTE PTR CNTDOWN      ; ELSE DECREMENT COUNTER
    JNZ  NO_FIN                ; IF COUNTER GOES TO ZERO, YOUR
                               ; DONE SO
    STC                        ; CY FLAG MUST BE SET.
    RET
NOTDON:
    MOV  BYTE PTR CNTDOWN,10       ; RESET COUNTER
    MOV  WORD PTR LASTDAC,BX       ; SAVE LAST OUTPUT TO DAC
    CLC                            ; CLEAR CY FLAG SINCE NOT
                                   ; DONE
NO_FIN:
    RET

CKDONE   ENDP
```

```
;
;*************************************************************
;
;  SUBROUTINE ENTRY POINT:  CTRLC
;  FUNCTION:      checks to see if ^C has been entered at
;                 keyboard
;                 if so, end program
;  INPUT:         none
;  OUTPUT:        none
;  REGISTERS:     all but AX are preserved
;  MODULES USED: none
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*************************************************************
;
CTRLC  PROC  NEAR
    CMP  AX,2E03H       ; CHECK BUFFER FOR ^C
    JNZ  NO_C           ; IF SO, END THE PROGRAM
    JMP  END
NO_C:
    MOV AH,0            ; ELSE TAKE CHAR FROM BUFFER
    INT  16H
    RET                 ; CONTINUE WITH MAIN ROUTINE
CTRLC   ENDP
```

```
;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  CHKDAC
;  FUNCTION:      make sure that the tare switch is set so
;                 that sample is lighter than counterweight
;  INPUT:         none
;  OUTPUT:        message to console if tare must be ;
;                 incremented
;  REGISTERS:     none are preserved
;  MODULES USED:  INPRT
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*****************************************************************
;
CHKDAC  PROC  NEAR
AGAIN:
    CALL INPRT                  ; GET INPUT FROM A/D
    AND  CX,8000H               ; CHECK IF NEGATIVE
    JZ   TAR_OK                 ; IF POSITIVE DISPLAY NOTHING
                                ; AND RETURN
    MOV  DX,OFFSET TARMSG       ; ELSE DISPLAY THE TARE MESSAGE
    MOV  AH,09                  ; DOS DISPLAY FUNCTION
    INT  21H                    ; DOS FUNCTION CALL
    JMP  AGAIN
TAR_OK:
    MOV  DX,OFFSET BLANK        ; SET POINTER TO DISPLAY BLANK
                                ; LINE
    MOV  AH,09                  ; DOS DISPLAY FUNCTION
    INT  21H                    ; DOS FUNCTION CALL
    RET
CHKDAC   ENDP
```

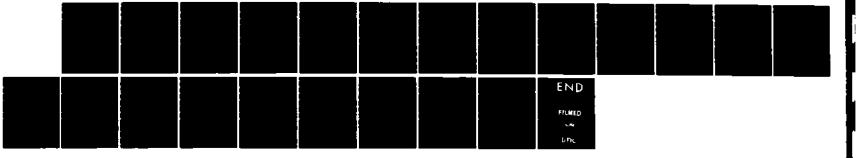
```
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  INPRT
;  FUNCTION:      gets input from ADC through I/O ports
;  INPUT:         none
;  OUTPUT:        CX : input from ADC
;  REGISTERS:     only BX is preserved
;  MODULES USED:  none
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*****************************************************************
;
INPRT    PROC     NEAR
    MOV  DX,279H          ; STATUS PORT ADDRESS
STAT:                ; WAIT FOR STATUS BIT HIGH
    IN   AL,DX            ; GET STATUS OF A/D
    AND  AL,00001000B     ; MASK OFF ALL BUT STATUS BIT #3
    JNZ  STAT             ; KEEP CHECKING UNTIL READY
STAT2:               ; WAIT FOR FALLING EDGE OF STATUS BIT
    IN   AL,DX
    AND  AL,00001000B
    JZ   STAT2

    MOV  DX,27AH          ; CONTROL FOR LOW ORDER BYTE OF
    MOV  AL,23H           ; A/D CONVERTER
    OUT  DX,AL            ;
    MOV  DX,278H          ; INPUT FROM A/D PORT
    IN   AL,DX
    MOV  CL,AL            ; SAVE INPUT IN C REGISTER
    MOV  DX,27AH          ; CONTROL FOR HIGH ORDER BYTE OF
    MOV  AL,29H           ; A/D CONVERTER
    OUT  DX,AL
    MOV  DX,278H          ; INPUT FROM A/D PORT AND
    IN   AL,DX            ; SAVE IN UPPER BYTE OF THE
    MOV  CH,AL            ; C REGISTER
    AND  CX,03FFFH        ; MASK OFF ANY NOISE THAT MAY HAVE
                          ; BEEN INPUTTED
    TEST CH,10H           ; TEST IF ADC IS OVERRANGED
    JZ   NO_OVF
    MOV  DX,27AH          ; IF OVERFLOW ,SOUND ALARM
    MOV  AL,25H
    OUT  DX,AL
NO_OVF:
    TEST CH,20H           ; CHECK FOR NEGATIVE INPUT
    JNZ  POSITV
    AND  CX,0FFFH         ; KEEP ONLY DATA BITS
    NEG  CX               ; IF NEG INPUT, NEGATE CX TO MATCH
                          ; INPUT
    RET
POSITV:
    AND  CX,0FFFH         ; KEEP ONLY DATA BITS
    RET
INPRT   ENDP
```

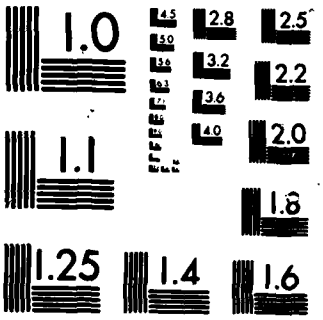
```
;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  GETOPT
;  FUNCTION:      determine output to DAC
;  INPUT:         CX : input from ADC
;  OUTPUT:        BX : output to DAC
;  REGISTERS:     none are preserved
;  MODULES USED: none
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*****************************************************************
;
GETOPT  PROC  NEAR
    CMP   CX,0000H        ; CHECK IF A/D INPUT IS ZERO
    JE    NOCHG           ; IF SO, LEAVE OUTPUT UNCHANGED
    JL    DECR            ; IF NEG INPUT, DECREMENT THE OUTPUT
    CMP   BX,0FFFH        ; ELSE INCR OUTPUT, UNLESS WE'RE AT
    JE    NOCHG           ; UPPER LIMIT OF THE DAC OUTPUT
    CMP   CX,0100H        ; IF INPUT IS GREAT, INCREMENT THE
    JGE   BIGINC          ; OUTPUT BY A FACTOR OF 10
    INC   BX              ; ELSE INCREMENT BY 1
    RET

DECR:
    CMP   BX,0000H        ; CHECK TO SEE IF DAC IS A LOWER
                          ; LIMIT
    JE    NOCHG           ; IF SO, DON'T DECREMENT
    CMP   CX,-100H        ; CHECK TO SEE IF INPUT MAGNITUDE IS
                          ; GREAT
    JLE   BIGDEC          ; IF SO, DECREMENT DAC BY A FACTOR
                          ; OF 10
    DEC   BX              ; ELSE DECREMENT BY 1
    RET

BIGDEC:
    SUB   BX,10           ; DECREMENT OUTPUT BY 10 AND RETURN
    RET

BIGINC:
    ADD   BX,10           ; INCREMENT OUTPUT BY 10 AND RETURN
    RET

NOCHG:
    RET
GETOPT  ENDP
```

```
;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  OUTPRT
;  FUNCTION:      output value to DAC
;  INPUT:         BX : input from ADC
;  OUTPUT:        value of BX to DAC I/O ports
;  REGISTERS:     only BX and CX are preserved
;  MODULES USED: none
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*****************************************************************
;
OUTPRT  PROC NEAR

    PUSH BX
    MOV  DX,378H        ; OUTPUT THE HIGH BYTE OF THE C REG
    MOV  AX,BX          ; TO THE D/A CONVERTER
    ROR  AX,1           ; HIGH BYTE OF DATA IS IN LOWER 4
                        ; BITS OF
    ROR  AX,1           ; CH AND UPPER 4 BITS OF CL.
                        ; THEREFORE, MUST ROTATE HIGH BYTE
    ROR  AX,1           ; OF DATA DOWN INTO THE AL REGISTER
    ROR  AX,1           ; AFTER MOVING CX TO AX.
    OUT  DX,AL
    MOV  DX,37AH        ; OUTPUT THE LOWER 4 BITS OF THE C
                        ; REG.
    MOV  AL,BL          ; TO THE D/A CONVERTER AND MASK OFF
    AND  AL,0FH         ; THE HIGHER 4 BITS
    OUT  DX,AL
    MOV  DX,27AH        ; CONTROL PORT FOR ENABLE ON LATCH
    MOV   AL,20H        ; TURN LATCH ON
    OUT  DX,AL
    POP  BX
    RET
OUTPRT    ENDP
```

AD-A164 014 DESIGN OF A DIGITAL CONTROLLER FOR AN ULTRAGRAVIMETRIC MICROBALANCE(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING C J BOLAN DEC 85 2/2

UNCLASSIFIED AFIT/GE/ENG/85D-4 F/G 14/2 NL





```
;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  DISPLY
;  FUNCTION:      DISPLAY VALUE IN CX ONTO CONSOLE
;  INPUT:         CX : CONTAINS VALUE TO BE DISPLAYED
;                 DX : CONTAINS POSITION ON SCREEN FOR DISPLAY
;  OUTPUT:        VALUE OF CX ON CONSOLE IN POSITION GIVEN BY
;                 DX REGISTER
;  REGISTERS:     ONLY BX IS PRESERVED
;  MODULES USED:  SHOW, CONVRT
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*****************************************************************
;
DISPLY  PROC NEAR
    CMP   CX,0              ; IF THE NUMBER TO BE DISPLAYED IS
    MOV   AL,' '            ; NEGATIVE, THEN DISPLAY A MINUS
    JGE   DSPSGN            ; SIGN IN FRONT OF THE NUMBER, ELSE
    NEG   CX                ; DISPLAY A BLANK
    MOV   AL,'-'            ;
DSPSGN:
    CALL  SHOW
    MOV   AL,CH             ; GET HIGHEST 4 BITS AND
    CALL  CONVRT            ; CONVERT TO ASCII
    CALL  SHOW              ; THEN DISPLAY CHARACTER VALUE
    MOV   AL,CL             ; GET MID 4 BITS AND CONVERT TO
    ROR   AL,1              ; ASCII AND DISPLAY
    ROR   AL,1
    ROR   AL,1
    ROR   AL,1
    CALL  CONVRT
    CALL  SHOW
    MOV   AL,CL             ; GET LOWEST 4 BITS AND
    CALL  CONVRT            ; CONVERT TO ASCII AND THEN
    CALL  SHOW              ; DISPLAY THE CHARACTER VALUE
    RET
DISPLY  ENDP
```

```
;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  CONVRT
;  FUNCTION:      CONVERT VALUE IN AL REG TO ASCII CODE
;  INPUT:         AL : CONTAINS VALUE TO BE CONVERTED TO ASCII
;  OUTPUT:        AX : CONTAINS ASCII VALUE
;  REGISTERS:     ALL BUT AX ARE PRESERVED
;  MODULES USED: SHOW, CONVRT
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*****************************************************************
;
CONVRT    PROC    NEAR
    AND   AL,0FH          ; MASK OFF HIGH BITS
    CMP   AL,09
    JBE   NEXT
    ADD   AL,07           ; ASCII ADJUST
NEXT:
    ADD   AL,'0'
    RET
CONVRT  ENDP

;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  SHOW
;  FUNCTION:      DISPLAY THE ASCII CHARACTER IN AL ONTO
;                 SCREEN
;  INPUT:         AL : CONTAINS ASCII VALUE TO BE DISPLAYED
;  OUTPUT:        ASCII CHAR DISPLAYED ON CONSOLE
;  REGISTERS:     ALL BUT AX ARE PRESERVED
;  MODULES USED: NONE
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*****************************************************************
;
SHOW  PROC   NEAR
    PUSH BX
    PUSH CX               ; SAVE THE CONTENTS OF THE C REG
    MOV  BH,00            ; PAGE 0 (CURRENT SCREEN)
    MOV  AH,2             ; PLACE THE CURSOR IN THE
    INT  10H              ; DESIGNATED POSITION
    INC  DX               ; INCREMENT COLUMN POINTER
    MOV  CX,1             ; EACH CHAR WILL BE REPEATED ONCE
    MOV  BX,000DH         ; CHAR WILL BE THE COLOR MAGENTA
    MOV  AH,9             ; FUNCTION #9 OF VIDEO INT VECTOR
    INT  10H              ; VIDEO INTERRUPT VECTOR
    POP  CX               ; RESTORE C REGISTER
    POP  BX
    RET
SHOW    ENDP
```

```
;
;*************************************************************
;
;  SUBROUTINE ENTRY POINT:  STORE
;  FUNCTION:      if one sec has passed, store value in BX
;                 onto data stack
;  INPUT:         BX : contains value to be stored
;  OUTPUT:        none
;  REGISTERS:     all but AX and DX are preserved
;  MODULES USED: none
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*************************************************************
;
STORE  PROC  NEAR
     PUSH BX
     PUSH CX
     MOV  AH,2CH                    ; GET TIME
     INT  21H
     CMP  DH,BYTE PTR STO_CNT       ; IF 1 SEC HASN'T PASSED,
     JNE  SKIP                      ; RETURN
     INC  WORD PTR TCOUNT           ; ELSE, INCREMENT TO NEXT
                                    ; SECOND
     INC  BYTE PTR STO_CNT
     XOR  CH,CH                     ; DISPLAY TIME ON CONSOLE
     MOV  DX,0323H                  ;
     MOV  CL,BYTE PTR TCOUNT        ;
     CALL DISPLY                    ;
     CMP  BYTE PTR STO_CNT,60       ; CHECK TO SEE IF COUNTER
                                    ; IS OVER
     JNE  NO_ADD                    ; 60 SECONDS. IF SO, MAKE
                                    ; COUNTER
     MOV  BYTE PTR STO_CNT,00       ; EQUAL TO ZERO SECONDS
NO_ADD:
     MOV  AX,BX                     ; GET DAC VALUE, SUBTRACT
                                    ; THE ZERO
     SUB  AX,WORD PTR ZRO_DAC       ; OFFSET, AND STORE
     MOV  BX,OFFSET ES:STACK        ; GET STARTING STACK ADDR
     ADD  BX,WORD PTR COUNTER       ; ADD COUNTER TO STACK
                                    ; ADDR TO GET THE
     MOV  ES:[BX],AX                ; CURRENT ADDR TO PLACE
                                    ; DATA IN MEMORY
     ADD  WORD PTR COUNTER,2        ; INCREMENT ADDRESS
                                    ; COUNTER BY 2 BYTES
     MOV  AX,WORD PTR COUNTER
     ROR  AX,1
     ROR  AX,1
     CMP  AX,3FFFH                  ; CHECK TO SEE IF AT END
                                    ; OF MEMORY
     JL   SKIP                      ; IF NOT KEEP GOING
     CALL DISK                      ; ELSE WRITE INFO TO DISK
                                    ; AND QUIT
     JMP  END                       ; IF SO, END
```

```
SKIP:
        POP   CX
        POP   BX
        RET
STORE   ENDP
```

```
;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  DISK
;  FUNCTION:      store data in stack in DATA SEG onto a disk
;                 file
;  INPUT:         none
;  OUTPUT:        disk file
;  REGISTERS:     none preserved
;  MODULES USED: none
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*****************************************************************
;
DISK     PROC  NEAR
     MOV  DX,OFFSET FNAME          ; GET FILE NAME
     MOV  CX,0
     MOV  AH,3CH                   ; FUNTION FOR CREATING A
                                   ; FILE
     INT  21H
     MOV  HANDLE,AX                ; HANDLE OF FILE
     JC   ERROR

     MOV  BX,HANDLE
     PUSH DS
     MOV  AX,ES
     MOV  DS,AX                    ; DS POINTS TO DATA STACK
     MOV  DX,OFFSET STACK
     MOV  CX,7FFFH                 ; WRITE 32 KBYTES TO FILE
     MOV  AH,40H                   ; FNCT FOR WRITING TO DISK
     INT  21H                      ; FUNCTION CALL INTERRUPT
     POP  DS
     JC   ERROR                    ; IF ERROR IN WRITING
                                   ; DISPLAY ERROR CODE
     CMP  AX,7FFFH
     JNE  ERROR
     CMP  WORD PTR TCOUNT,7FFFH    ; SEE IF MORE DATA NEEDS
                                   ; WRITTEN
     JLE  EXIT                     ; IF NOT, DONE
     PUSH DS
     MOV  AX,ES
     MOV  DS,AX                    ; WRITE LAST HALF OF DATA
                                   ; TO DISK
     MOV  DX,OFFSET STACK + 7FFFH
     MOV  AH,40H                   ; FNCT FOR WRITING TO FILE
     MOV  CX,WORD PTR TCOUNT - 7FFFH
     INT  21H                      ; FUNCTION CALL INTERRUPT
     POP  DS
     JC   ERROR                    ; IF ERROR, DISPLAY ERROR
                                   ; CODE
     CMP  AX,WORD PTR TCOUNT - 7FFFH
     JNE  ERROR
```

```
EXIT:
    CLC                             ; CLOSE DISK FILE
    MOV  BX,HANDLE
    MOV  AH,3EH                     ; FUNCTION FOR CLOSING A
                                    ; FILE
    INT  21                         ; FUNCTION CALL
    JC   ERROR                      ; IF ERROR IN CLOSING,
                                    ; DISPLAY ERROR
    RET

ERROR:
    MOV  BX,AX                      ; DISPLAY ERROR CODE ON
                                    ; CONSOLE
    MOV  DX,0123H
    CALL DISPLY
    JMP  END

DISK    ENDP
;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  END
;  FUNCTION:      returns to control to operating system
;  INPUT:         none
;  OUTPUT:        none
;  REGISTERS:     none are preserved
;  MODULES USED:  none
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*****************************************************************
;
END:
    MOV  AH,4CH          ; FUNCTION FOR RETURNING TO DOS
    XOR  AL,AL
    INT  21H
```

```
;
;*****************************************************************
;  D A T A
;*****************************************************************
;
PRMSG    DB    0DH,0AH,'TYPE ANY KEY TO CONTINUE','$'
TARMSG   DB    0DH,'   DECREMENT TARE SWITCH OR SAMPLE TOO'
         DB    'LIGHT','$'
BLANK    DB    0DH,'                                              '
         DB    '     ','$'
STO_CNT  DB    00H              ; SECONDS COUNTER
         DB    00H              ; THIS IS JUST FOR TESTING
                                ; PURPOSES
FNAME    DB    'DATA.TXT',0
HANDLE   DW    ?
COUNTER  DW    0000H            ; COUNTER FOR MEMORY LOCATION
                                ; FOR DATA
TCOUNT   DW    0000H            ; SECONDS COUNTER FOR STORAGE
                                ; OF DATA
ZRO_CNT  DB    00H              ; COUNTER FOR # TIMES INPUT IS
                                ; ZERO
ZRO_DAC  DW    ?                ; THE ZERO OFFSET FOR THE
                                ; BALANCE
LASTDAC  DW    ?
CNTDOWN  DB    10
CODE ENDS

;*****************************************************************
;
; THIS IS THE LOCATION OF THE DATA STACK
; FOR THE OUTPUT VALUES
;
;*****************************************************************
DATA SEGMENT

STACK    DB    0FFFFH DUP(11H)

DATA     ENDS
    END
```

APPENDIX F: HARDWARE TESTING ROUTINES

ANALOG-TO-DIGITAL CONVERTER TESTING ROUTINE

```
CODE SEGMENT PUBLIC
ASSUME CS:CODE
;*****************************************************************
;
;    PROGRAM NAME: ADC TEST ROUTINE
;    VERSION:      1.0
;    SYSTEM:       MSDOS (Zenith 151)
;    LANGUAGE:     8088 Assembler (MASM)
;    AUTHOR:       2Lt Chris Bolan
;    DATE:         7 July 1985
;
;    This program tests a 12-bit analog-to-digital converter
;    and its interface with a Zenith 151 Computer.  The
;    routine takes the output from the converter and displays
;    it on the console.
;
;*****************************************************************
;
    ORG  100H
MAIN:
    MOV  DX,27AH        ; INITIALIZE PORT 278H TO BE AN
    MOV  AL,20H         ; INPUT PORT BY SETTING D5 OF 27AH
    OUT  DX,AL

    MOV  DX,3D8H        ; SCREEN IS 80 X 25 TEXT MODE
    MOV  AL,29H
    OUT  DX,AL

    INC  DX             ; MAKE BORDER BLUE OF CONSOLE
    MOV  AL,09
    OUT  DX,AL

    MOV  AX,0600H       ; CLEAR SCREEN
    MOV  BH,7           ; USE BLACK AS CLEARED COLOR
    MOV  CX,0           ; UPPER ROW, COL OF SCREEN
    MOV  DX,184FH       ; LOWER ROW, COL OF SCREEN
    INT  10H

CONTIN:
    CALL INPRT          ; GET INPUT FROM A/D CONVERTER
    CALL DISPLY         ; PRINT VALUE TO THE SCREEN

    MOV  AH,01          ; CHECK KEYBOARD STATUS
    INT  16H            ; KEYBOARD INTERRUPT
    JNZ  CTRLC          ; IF KEYBRD ENTRY CHECK FOR ^C

    JMP  CONTIN
```

```
;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  INPRT
;  FUNCTION:      gets input from ADC through I/O ports
;  INPUT:         none
;  OUTPUT:        CX : input from ADC
;  REGISTERS:     none are preserved
;  MODULES USED: none
;  VERSION:       1.0 - 7 JulY 85 - C. BOLAN
;
;*****************************************************************
;
INPRT  PROC  NEAR
    MOV  DX,279H          ; STATUS PORT ADDRESS
STAT:
    IN   AL,DX            ; GET STATUS OF A/D
    AND  AL,08            ; MASK OFF ALL BUT STATUS BIT #3
    JNZ  STAT             ; KEEP CHECKING UNTIL READY

    MOV  DX,27AH          ; CONTROL FOR LOW ORDER BYTE OF
    MOV  AL,28H           ; A/D CONVERTER
    OUT  DX,AL            ;
    MOV  DX,278H          ; INPUT FROM A/D PORT
    IN   AL,DX
    MOV  CL,AL            ; SAVE INPUT IN C REGISTER
    MOV  DX,27AH          ; CONTROL FOR HIGH ORDER BYTE OF
    MOV  AL,22H           ; A/D CONVERTER
    OUT  DX,AL
    MOV  DX,278H          ; INPUT FROM A/D PORT AND
    IN   AL,DX            ; SAVE IN UPPER BYTE OF THE
    MOV  CH,AL            ; C REGISTER
    AND  CX,03FFFH        ; MASK OFF ANY NOISE THAT MAY HAVE
                          ; BEEN INPUTTED
    RET
INPRT   ENDP
```

```
;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  CTRLC
;  FUNCTION:      checks to see if ^C has been entered at
;                 keyboard.  If so, end program
;  INPUT:         none
;  OUTPUT:        none
;  REGISTERS:     all but AX are preserved
;  MODULES USED: none
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*****************************************************************
;
CTRLC   PROC NEAR
    CMP   AX,2E03H        ; CHECK BUFFER FOR ^C
    JZ    END             ; IF SO, END THE PROGRAM
    MOV   AH,0            ; ELSE TAKE CHAR FROM BUFFER
    INT   16H
    JMP   CONTIN          ; CONTINUE WITH MAIN ROUTINE
CTRLC   ENDP
```

```
;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  DISPLY
;  FUNCTION:    display value in CX onto console
;  INPUT:       CX : contains value to be displayed
;               DX : contains position on screen for display
;  OUTPUT:      value of CX on console in position given by
;               DX
;  REGISTERS:   only BX is preserved
;  MODULES USED: SHOW, CONVRT
;  VERSION:     1.0 - 7 July 85 - C. BOLAN
;
;*****************************************************************
;
DISPLY  PROC NEAR
    MOV  DX,0D23H       ; SCREEN POSITION
    MOV  AL,CH          ; GET HIGHEST 4 BITS AND
    CALL CONVRT         ; CONVERT TO ASCII
    CALL SHOW           ; THEN DISPLAY CHARACTER VALUE
    MOV  AL,CL          ; GET MID 4 BITS AND CONVERT TO
    ROR  AL,1           ; ASCII AND DISPLAY
    ROR  AL,1
    ROR  AL,1
    ROR  AL,1
    CALL CONVRT
    CALL SHOW
    MOV  AL,CL          ; GET LOWEST 4 BITS AND
    CALL CONVRT         ; CONVERT TO ASCII AND THEN
    CALL SHOW           ; DISPLAY THE CHARACTER VALUE

    RET
DISPLY  ENDP
```

```
;
;*************************************************************
;
;  SUBROUTINE ENTRY POINT:  CONVRT
;  FUNCTION:      convert value in AL reg to ASCII code
;  INPUT:         AL : contains value to be converted to ASCII
;  OUTPUT:        AX : contains ASCII value
;  REGISTERS:     all but AX are preserved
;  MODULES USED: SHOW, CONVRT
;  VERSION:       1.0 - 13 NOV 85 - C. BOLAN
;
;*************************************************************

;
CONVRT  PROC NEAR
    AND  AL,0FH          ; MASK OFF HIGH BITS
    CMP  AL,09
    JBE  NEXT
    ADD  AL,07           ; ASCII ADJUST
NEXT:
    ADD  AL,'0'
    RET
CONVRT   ENDP
```

```
;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  SHOW
;  FUNCTION:      display the ASCII character in AL onto
;                 screen
;  INPUT:         AL : contains ASCII value to be displayed
;  OUTPUT:        ASCII char displayed on console
;  REGISTERS:     all but AX are preserved
;  MODULES USED: none
;  VERSION:       1.0 - 7 July 85 - C. BOLAN
;
;*****************************************************************
;
SHOW   PROC  NEAR
     PUSH CX          ; SAVE THE CONTENTS OF THE C REG
     INC  DX          ; INCREMENT COLUMN POINTER
     MOV  AH,2        ; PLACE THE CURSOR IN THE
     INT  10H         ; DESIGNATED POSITION

     MOV  CX,1            ; EACH CHAR WILL BE REPEATED ONCE
     MOV  BX,000DH ; CHAR WILL BE THE COLOR MAGENTA
     MOV  AH,9            ; FUNCTION #9 OF VIDEO INT VECTOR
     INT  10H         ; VIDEO INTERRUPT VECTOR
     POP  CX          ; RESTORE C REGISTER
     RET
SHOW       ENDP
;

END:
     MOV  AH,4CH    ; FUNCTION FOR RETURNING TO DOS
     XOR  AL,AL
     INT  21H

CODE   ENDS
       END     MAIN
```

```
DIGITAL-TO-ANALOG CONVERTER TESTING ROUTINE

CODE SEGMENT PUBLIC
ASSUME CS:CODE
;*****************************************************************
;
;    PROGRAM NAME: DAC TEST ROUTINE
;    VERSION:      1.0
;    SYSTEM:       MSDOS (Zenith 151)
;    LANGUAGE:     8088 Assembler (MASM)
;    AUTHOR:       2Lt Chris Bolan
;    DATE:         7 July 1985
;
;    This program tests a 12-bit digital-to-analog converter
;    and its interface with a Zenith 151 Computer.  The
;    routine increments the output to the DAC every time a
;    there is a keyboard entry (except for ^C).
;
;*****************************************************************
;

    ORG  100H
MAIN:
    MOV  CX,00
LOOP:
    MOV  AH,01          ; CHECK KEYBOARD STATUS
    INT  16H            ; KEYBOARD INTERRUPT
    CALL OUTPRT
    JMP  LOOP
```

```
;
;*****************************************************************
;
;  SUBROUTINE ENTRY POINT:  OUTPRT
;  FUNCTION:      output value to DAC
;  INPUT:         none
;  OUTPUT:        value of CX to DAC I/O ports
;  REGISTERS:     only CX is preserved
;  MODULES USED:  none
;  VERSION:       1.0 - 7 July 85 - C. BOLAN
;
;*****************************************************************
;
OUTPRT:
    CMP   AX,2E03H        ; CHECK BUFFER FOR ^C
    JZ    END             ; IF SO, END THE PROGRAM
    MOV   AH,0            ; ELSE TAKE CHAR FROM BUFFER
    INT   16H

    INC   CX
    MOV   DX,37AH         ; OUTPUT THE LOW BYTE OF CX
    MOV   AL,CL           ; TO THE D/A CONVERTER
    OUT   DX,AL
    MOV   DX,378H         ; OUTPUT THE HIGHER BYTE OF CX
    MOV   AX,CX           ; TO THE D/A CONVERTER
    ROR   AX,1
    ROR   AX,1
    ROR   AX,1
    ROR   AX,1
    OUT   DX,AL
    MOV   DX,27AH         ; CONTROL PORT FOR ENABLE ON LATCH
    MOV   AL,20H          ; TURN LATCH ON
    RET
;
;
END:    NOP

CODE    ENDS
    END   MAIN
```

Bibliography

1. Angell, C.L., and Czanderna, A.W. "The Simultaneous Measurement of Mass Change and Optical Spectra," Ultra Micro Weight Determination in Controlled Environments, edited by S.P. Wolsky and E.J. Zadnuk. New York: John Wiley and Sons, Inc., 1969.

2. Czanderna, A. W. "Ultramicrobalance Review," Ultra Micro Weight Determination in Controlled Environments, edited by S. P. Wolsky and E. J. Zadnuk. New York: John Wiley and Sons, Inc., 1969.

3. Czanderna, A.W. and Honig, J.M. "Sensitive Quartz Beam Microscope," Analytical Chemistry, 29(8): 1206 - 10, August 1957.

4. Czanderna, A. W. and Wolsky, S.P. "Introduction and Microbalance Review," Microweighing in Vacuum and Controlled Environments, edited by A. W. Czanderna and S. P. Wolsky. New York: Elsevier Scientific Publishing Company, 1980.

5. Czanderna, A. W., et al. "Photoelectrically Automated, Bakeable, High-load Ultramicrobalance," Journal of Vacuum Science Technology, 13(1): 556-59 (January/February 1976).

6. D'azzo, J.J, and Houpis, C.H. Linear Control System Analysis and Design. New York: Mcgraw Hill, 1975.

7. Intersil. "ICL7109 12 Bit Binary A/D Converter for Microprocessor Interfaces," Hot Ideas in CMOS. Data Sheet Book. Intersil Inc., 1984.

8. Johnson, Hilburn, and Johnson. Basic Electric Circuit Analysis. New Jersey: Prentice Hall, 1978.

9. Schwobel, R. L. "Beam Microbalance Design, Construction and Operation," Microweighing in Vacuum and Controlled Environments, edited by A. W. Czanderna and S. P. Wolsky. New York: Elsevier Scientific Publishing Company, 1980.

10. Thomas, III, J.H., and Sharma, S.P. Adsorption and desorption of water on Au by the quartz crystal-oscillator method," Journal of Vacuum Science Technology, 13(1): 549 - 551, (January/February 1976).

11. Vasofsky, R., Czanderna, A.W., and Thomas, R.W. "UHV System for the Ultramicrogravimetric Study of Samples Loaded in a Controlled Environment," Journal of Vacuum Science Technology, 16(2): 711-14 (March/April 1979).

12. Windisch, Charlie. Personal Communications. RADC/RBRE, Griffiss AFB NY, September 1979.

13. -----. Apparat Combo II Card Hardware Reference Manual, Installation Guide, and PC Utility Disk Manual. Product Instruction Manual. Apparat, Inc., Denver, CO., 4 November 1983.

14. -----. High Accuracy Data Acquisition Instrumentation Amplifier: AD524. Product Brochure. Analog Devices, Norwood, Massachusetts, 1985.

15. -----. HSDAC87: Complete 12-Bit D/A Converter. Product Brochure. Hybrid Systems Corp., 1985

16. -----. Programmer's Reference Manual: Z-100 PC series Computers. Zenith Data Systems Corporation, 1984.

VITA

Second Lieutenant Christopher J. Bolan was born on 24 September 1962 in Santa Monica, California. He graduated from high school from Apple Valley, California in 1980 and attended Loyola Marymout University in Los Angeles from which he received the degree of Bachelor of Science in Electrical Engineering in May 1984. Upon graduation, he received a commission in the USAF through the ROTC program. He was immediately called to active duty and entered a Master of Science in Electrical Engineering program at the School of Engineering, Air Force Institute of Technology, in June of 1984.

Permanent address: 13720 Lakota Road
Apple Valley, CA 92307

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

AD-A164 014

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): AFIT/GE/ENG/85D-4	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: School of Engineering; 6b. OFFICE SYMBOL (If applicable): AFIT/ENG	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State and ZIP Code): Air Force Institute of Technology, Wright-Patterson AFB, Ohio 45433	7b. ADDRESS (City, State and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: Product Evaluation Section; 8b. OFFICE SYMBOL (If applicable): RADC/RBRE	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State and ZIP Code): Rome Air Development Center, Griffiss AFB, New York 13441	10. SOURCE OF FUNDING NOS. — PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT NO.
11. TITLE (Include Security Classification): See Box 19	
12. PERSONAL AUTHOR(S): Christopher J. Bolan, B.S., 2d Lt, USAF	
13a. TYPE OF REPORT: MS Thesis; 13b. TIME COVERED: FROM _____ TO _____; 14. DATE OF REPORT (Yr., Mo., Day): 1985 December; 15. PAGE COUNT: 113	
16. SUPPLEMENTARY NOTATION	

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB. GR.	Digital Controls, Computer Hardware, Programming
09	02		
09	03		

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Title: DESIGN OF A DIGITAL CONTROLLER FOR AN ULTRAGRAVIMETRIC MICROBALANCE

Thesis Advisor: Donald R. Kitchen, Major, USAF

Approved for public release: IAW AFR 190-1/.
LYNN E. WOLAVER 16 JAN 86
Dean for Research and Professional Development
Air Force Institute of Technology (AFIT)
Wright-Patterson AFB OH 45433

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT: ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL: Donald R. Kitchen, Major, USAF	22b. TELEPHONE NUMBER (Include Area Code): 513-255-3576; 22c. OFFICE SYMBOL: AFIT/ENG

DD FORM 1473, 83 APR — EDITION OF 1 JAN 73 IS OBSOLETE.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

The design and construction of a digital control system for an Ultragravimetric pivotal beam microbalance has been performed. The digital system replaces the present analog control circuitry of the microbalance. Included in the design is a hardware circuit, a Z-151 computer which has been interfaced to the circuit, and software to control the circuit and to operate the microbalance.

An analysis of the present analog control circuit is first presented. This analysis gives the overall function of the circuit as well as an in-depth view of the configuration of a particular operational amplifier: the derivative compensator. The hardware for the digital control system is described followed by the development of the software which controls the system.

Studies performed on the experimental design revealed that the system satisfactorily simulates the analog control circuit. Tests show that the system automatically zeroes the balance, weighs the sample, stores the data onto a disk file, and terminates the experiment

END

FILMED

3-86

DTIC